Universität Leipzig
Fakultät für Mathematik und Informatik
Abteilung Datenbanken

A comparison of HTML-aware tools for Web Data extraction

Diplomarbeit

Leipzig, September, 2008

vorgelegt von

Xavier Azagra Boronat
Master-Studiengang Informatik

Betreuender Hochschullehrer: Prof. Dr. Erhard Rahm
Betreuer: Dr. Andreas Thor

Index

1- Introduction

Nowadays we live in a world where information is present everywhere in our daily life. In those last years the amount of information that we receive has grown and the stands in which is distributed have changed; from conventional newspapers or the radio to mobile phones, digital television or the Web. In this document we reference to the information that we can find in the Web, a really big source of data which is still developing.

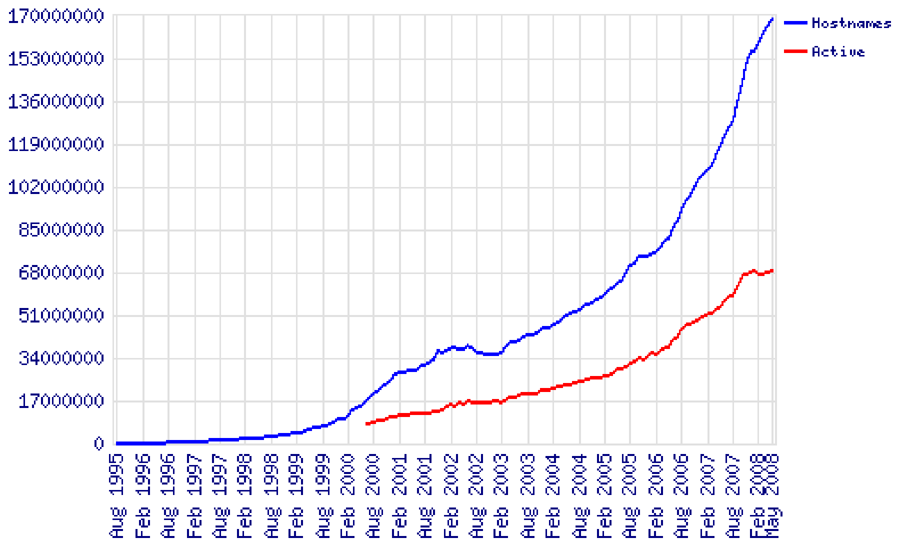


Figure 1: Growth of the number of hostnames – from [7]

Figure 1 illustrates the growth of hostnames in the last years. As shown, the curve represents a kind of exponential form and that means that the growth tendency is going to increase further. The same happens when talking about Wikipedia, an online and free encyclopedia. The number of articles is increase equivalent this exponential form, and with other words, more and more information is inserted into the Web.

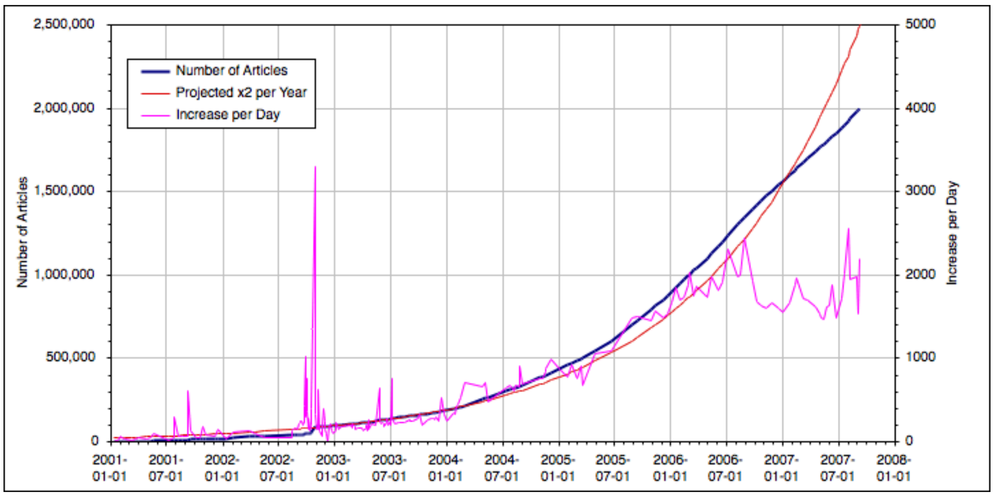


Figure 2: Growth of the number of Wikipedia articles – from [10]

In particular we are going to concentrate on the information we find in Web pages. Those had evolved introducing dynamic content: animations, video, visual content, audio, etc… One of the main problems that we are faced with is how to structure this big amount of information. At the beginning, the Web was designed as a source of data for a human use. It was built to guarantee that the content and the information could easily be understood and read by humans but not prepared to be used as data able to be treated by other applications. Because of this fact this kind of representation is not the most appropriate to extract data and sometimes we have to deal with difficulties.

When talking about the use of information, possibly this data will not be useful for any profile of user or its excess could produce information saturation or what is more maybe we are only interested in a particular share. On the other hand it could be useful to transform this information to deal with it later or use it in other areas.

This is where the data extraction process takes importance. Specific data is able to be extracted from all these Web sources in order to be used by other users or applications. The capacity to get specific information lets realize a summarization of this big amount of data located in the Web and use it for concrete purposes. So the importance of the Web data extraction resides on the fact that we realize extractions of all the content. At the same time such extraction presents problems when considering the data that we want to extract and how we realize this extraction.

One possibility is a manual extraction of this data but it is not viable because of the big amount of information that we have to treat with. We have to find another solution of this problem. Several data extraction tools have already been developed as a solution to this fact. They are specialized programs that can extract data in a manual, a semi-automatic or automatic way. They use the structure of the data sources and that give a final output of the extracted data.

We are going to use a set of tools that have been specifically designed for this purpose. First we will explain the data extraction process and then we will characterize each of these tools and execute several tests in some constructed scenarios. The main motivation of this document is to realize a categorization of the tools explaining the weak and strong points of them. We will find out which of them is suitable for different scenarios.

2- Data extraction process

In this chapter we are going to explain the data extraction process used to achieve data extractions. This is really significant as here we are going to explain how all this process works and which the possibilities to extract data are.

We are going to detail each of the aspects of the data extraction process, from the main purposes of Web data extraction to the main problems that can be found when performing extractions.

Mainly the fact of including this chapter in the document let the reader to have an overview of the situation of the data extraction process. We will talk about querying data on the Web and an idea of what is a wrapper, the selected method to extract data. Somehow we pretend to make easier the understanding of some characteristics of the data extraction when performing our tests and our final tool categorization introducing the main problems and the main techniques to extract data.

2.1- Characteristics of the data extraction process

Nowadays we can find several services and tools based on data extracting techniques for end-users that allow them to extract information from Web pages. The process of extracting structured data from Web sites is not a trivial task as most of the information is formatted in the Hypertext Markup Language (HTML). Knowing that, this format is designed for presentation purposes and not for automated data extraction. It happens that most of the HTML content of the Web is semi-structured. It means that pages with this type of content are in an intermediate position between structured and unstructured format and they don’t conform to a description for the types of data published therein. This situation is unlikely to change in short or even medium term for at least two reasons: the simplicity and power of HTML authoring tools together with a valuable inertia to change markup language.

A vast quantity of semi-structured data stored in electronic form is not present in HTML pages but in text files, such as e-mails, program code, documentation, configuration files… Therefore it is very important that some data extraction tools might be able to extract this kind of information.

However, in real-life scenarios data extraction capabilities are only one half of the game. We can find password-protected sites, cookies, Sessions IDs, JavaScript or dynamic changes on Web sites that make Web data extraction difficult in real-life application scenarios.

Two of the most important purposes when talking about Web data extraction are:

- Information retrieval (e.g. feeds, Web search engines, information services…)
- Economical issues (e.g. stock market, shopping comparison…)

In order to perform Web data extractions we are going to use a set of tools designed for this purpose. Normally to specify the input we provide our tools one or more Web page sources. The most common way to access the information is by giving the URL where these Web pages are located. Otherwise some tools can directly take a path to a file and extract its data. Once the tool knows where the source information is, its users work to configure the data extraction process.

About the data output, we can find several formats depending on the used tool. The most common formats of the extracted data are XML, HTML, RSS/ATOM Feeds or plain text, being XML the most used. Some tools are designed to directly transform the extracted data to other more specific Web formats, such as modules for Web portals or proprietary formats. Possible options that some of the tools present are putting the extracted data and embedding it on a Flash object or send it directly per email.

The data extraction is only a step when speaking about the process of getting data from the Web. This data is queried by human users or by applications, in this case we access to the stored data of other computers. This data could be stored in files, in databases or directly in HTML documents. When a user performs a standard query it uses a Web browser to access directly to the HTML Web data sources.

Another possibility is to realize an extraction process when we want to extract concrete information of the sources, and then an integration process when we retrieve information from more than one data source. This last process is responsible for joining the information in order to deal with unified data.

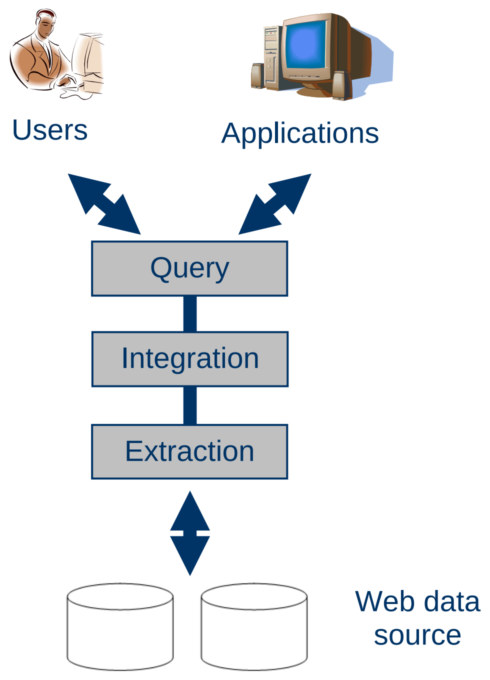


Figure 3: Querying data from the Web

A classification when speaking about the structure type of the data exists:

- Free text: This type of text could be found in natural language texts, for example magazines or pharmaceutical research abstracts. Patterns involving syntactic relations between words or semantic classes of words are used to extract data from this type of sources.

- Structured text: This type of text is defined as textual information in a database or file following a predefined and strict format. To extract this kind of data we have to use the format description.

- Semi-structured text: This type of text is placed in an intermediate point between unstructured collections of textual documents and fully structured tuples of typed data. To extract data we use extraction patterns that are often based on tokens and delimiters, for example the HTML-tags.

We are going to explain the various possibilities to extract information and how they work as follows. Basically, there exist three different ways to perform the extraction:

- Manual extraction of the data
- Use a built API
- Use a (semi)automatic wrapper

Manual extraction is the most precise option to extract data as we directly choose the data fields of our interest. The necessity to treat elements in an individual way takes a lot of time when treating large amount of data and hence makes to rule out this option as it is not viable. This could be a good option for small and concrete data extractions. However, it is not the most common scenario when talking about Web data extractions. For these reasons, these extractions should be performed in a more automatically way.

On the other hand an API belongs to the owner of the Web page where we want to extract data. Normally, we can find APIs in few specific numbers of Web pages and its use and supply are limited by the specifications of the owner. To use them we have to take a look at the documentation and the method list of the owner.

A wrapper let the end-user use a set of methods without the necessity to have support of the owner of the Web page and with independence of the content. It can be seen as a procedure that is designed for extracting content of a particular information source for delivering the content of interest in a self-describing representation. Its target should be converting information implicitly stored as an HTML document into information explicitly stored as a data-structured for further processing. Due to these characteristics, we are going to choose this kind of tools to perform data extractions from the Web. A wrapper for a Web source accept queries about information in the pages of that source, fetches relevant pages from the source and extracts the requested information and returns the result.
The construction of a wrapper can be done manually or by using a semi-automatic or automatic approach. The manual generation of a wrapper involves the writing of ad-hoc code. The creator has to spend quite some time understanding the structure of the document and translating it into program code. The task is not trivial and hand coding could be tedious and error-prone. On the other hand, semi-automatic wrapper generation benefits from support tools to help design the wrapper. By using a graphical interface the user can describe which the important data fields to be extracted are. A specific configuration of the wrapper should be done for each Web page source as the content structure varies from each other. Expert knowledge in wrapper coding is not required at this stage, and it is also less error-prone that coding. On the other hand, the automatic wrapper generation uses machine-learning techniques, and the wrapper research community has developed learning algorithms for a spectrum of wrappers. This kind of wrapper require a minimum intervention of human experts and systems which go through a training phase, where it is fed with training examples, and, in many cases, this learning has to be supervised.

Generally, the steps to extract information using a wrapper are the following:

- Load the information of the source page
- Transform the source page for its posterior treatment
- Identify the appearing elements
- Filter these elements
- Export of the final data to an output format

The first and last steps are common to all types of wrappers as we need a data input and a data output to perform a data extraction.

Depending of the used wrapper type the intermediate steps could vary. We can find several types of wrappers following the taxonomy of [24]. This taxonomy is based on the main technique used by the tool to generate a wrapper, what led us to the following groups of tools: Languages for Wrapper Development, HTML-aware tools, NLP-based tools and Ontology-based tools. More details can be found in the paper.

Of all of these tool kinds we are going to center us in the most modern and practical, in fact HTML-aware tools.

2.2- Representation of Web page elements

As explained before most of the Web pages follow the HTML syntax independent of their content (images, Flash, scripts…). The main elements that construct the structure are the HTML tags. They are identified by a name and can contain attributes and inner content. For the correctness of its usage there is an already defined syntax that defines an order of appearance, which are the available attributes, wether a tag should have a close tag or not... An already created standard by W3C exists that exposes all the construction rules of HTML (http://www.w3.org/TR/html401)

Thanks to this structure, wrappers are able to detect the elements of a Web site and extract the desired information. They can recognize repetition patterns of tags to extract similar content, read the attributes of this tags to associate elements or extract elements in an individual way.

When specifically speaking about HTML-aware tools, before performing the extraction process, these tools turn the document into a parsing tree, a representation that shows its HTML tag hierarchy.

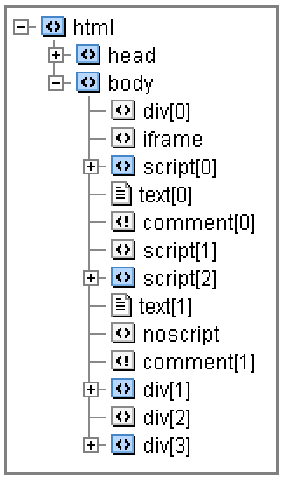


Figure 4: HTML parsing tree

Following, extraction rules are generated either semi-automatically or automatically and applied to the tree. In this three each node represents a tag while the outer tags are leaves. A specific tag is represented by a unique node and we can perform an expression to navigate through all the hierarchy, for example:

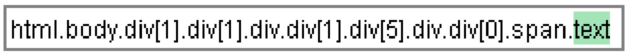

html.body.div[1].div[1].div.div[1].div[5].div.div[0].span.text

Figure 5: XPath expression to navigate through the HTML hierarchy

Due to this structure, the maximum precision to extract information from a Web page is found in the content of a leave. Afterwards, depending of the extracting data tool functionalities by treating the content more level of precision could be obtained.

All the content placed in a tag is suitable to be extracted. We can differentiate these tags by the identifier, the style of the tag when we use CSS, the tag attributes... This information is used by the data extraction tools to realize an extraction. Depending of the tool we have to proceed in a specific way to realize a good configuration of the wrapper.

2.3- HTML problems to extract data

As HTML has semi-structured content we can find some problems in the structure that could produce errors at the time of extracting data. These errors can be categorized in several groups. We are now going to comment each of them:

- **Presentation of the data without following a structure**

  Normally the content of a Web page is presented following structured patterns. This structure supplies the user an easy and logic way to find the information avoiding to waste his time. A good structure helps the data extraction tools to realize a good work.

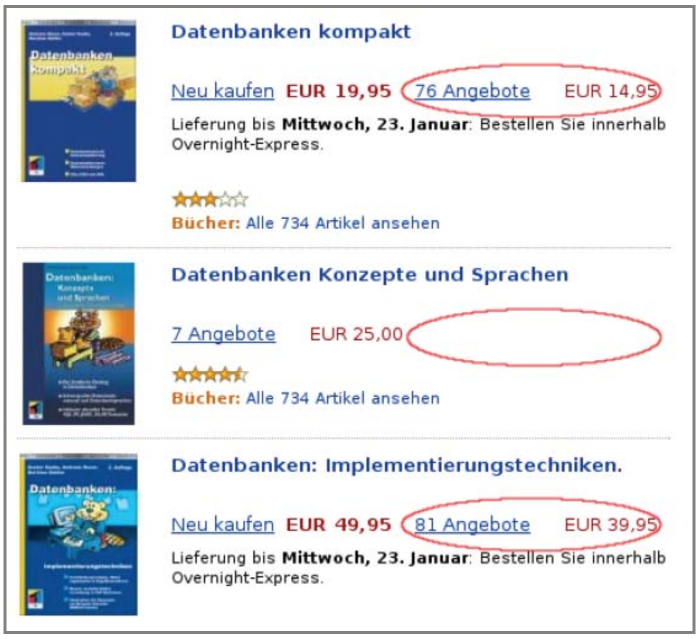
  A suitable example could be a scenario of a digital newspaper. In this scenario we can find a table that contains all the news ordered by time of success. Each row is composed by a headline and a brief description of the news. This way to structure is simple and if we represent it on a tree. We will see that some elements are appearing repeatedly. This will help our tools to extract the information. Let is imagine the opposite example, a digital newspaper that doesn’t use a main table with all the news and it doesn’t follow a rule to present the information. It means some news could have photos, others videos and the information will be presented in a cell of a specified size and location that makes a nice end-view to the user. This kind of structure has more possibilities to generate problems to our data extraction tools.

- **Bad constructed HTML source documents**

  A well-built HTML document must follow some rules. Although most of the browsers could visualize the content of a page having some errors in the structure it is highly recommended to follow the W3C standard of HTML. Some of these errors could consist of bad placed tags, repeated tags without sense, no closed tags... All these kind of mistakes could make harder our data extraction.

- **Nested data elements**

  These kinds of elements nest data and then element by element could appear differences. An example is shown on Figure 6.


Datenbanken kompakt

Neu kaufen **EUR 19,95** 76 Angebote EUR 14,95

Lieferung bis **Mittwoch, 23. Januar**: Bestellen Sie innerhalb Overnight-Express.

Bücher: Alle 734 Artikel ansehen

Datenbanken Konzepte und Sprachen

7 Angebote EUR 25,00

Bücher: Alle 734 Artikel ansehen

Datenbanken: Implementierungstechniken.

Neu kaufen **EUR 49,95** 81 Angebote EUR 39,95

Lieferung bis **Mittwoch, 23. Januar**: Bestellen Sie innerhalb Overnight-Express.

Figure 6: Example of nested data elements - from [27]

We want to extract the part of information that is related to the auction. What happens here is that the second element is not new and then this type of information is displaced to the beginning, this will produce errors. Similar examples of this kind could be found on the Web.

- **Problems choosing the correct Web page source example**

This problem can be shown choosing a Web page which content structure could change depending on some factors. One real example of this kind is the resulting page of Web search engines. If we perform a search using an input value we get a result page with some entries. Depending of this value, this resulting Web page will change. We would we get some image snapshots, video snapshots or some advertising related to this value. If the structure changes depending of this value, we can not use our data extraction tool with all the possible values to be sure it uses exactly the best source. Because of this fact, we can say it is really important in this kind of pages to select a good sample to assure that we are going to produce the minimum number of errors during the data extraction process.

- **Problems using scripts or dynamic content**

Our data extraction tools read the HTML code to perform extractions. All the static content is written in HTML, it doesn't occur when speaking about dynamic content; such as Javascript, AJAX or Flash. Our data extraction tools cannot parse or treat all this information like with normal HTML. It doesn't follow the same syntax, sometimes it has to be preprocessed before displaying a result or others the result is only visual or changes could be introduced at any time the page is loaded. Some of our tools have support to treat dynamic content, especially Javascript, but often this kind of content generates difficulties to perform data extractions.

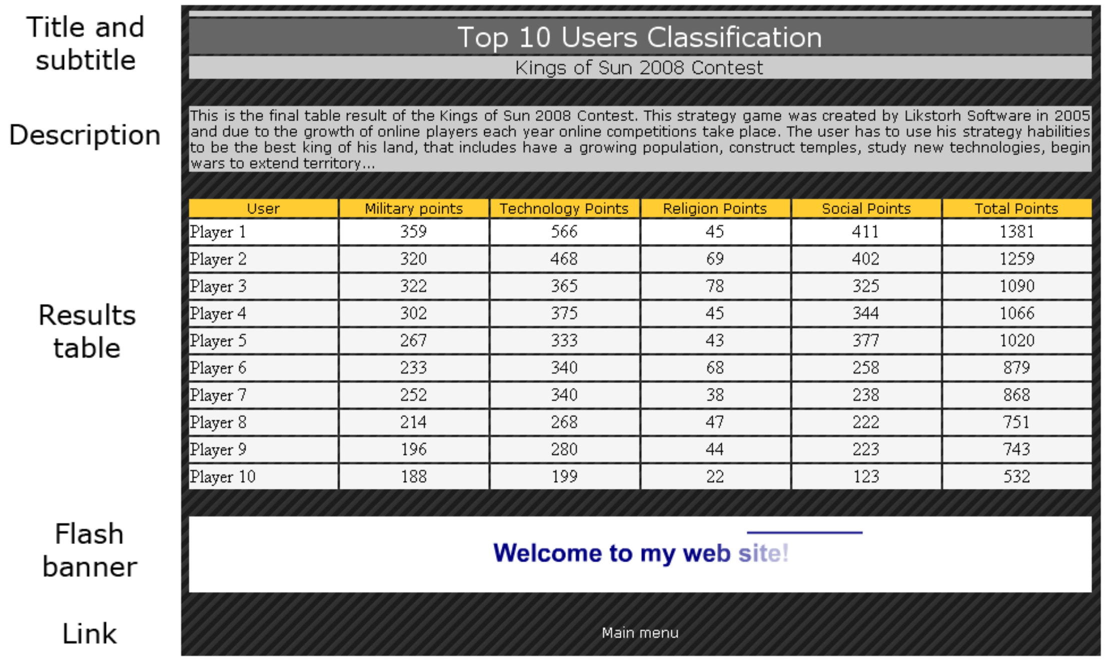
2.4- Ideal characteristics for a Web page to extract data: An example

Once analyzed the characteristics of the extraction process and the problems that could generate difficulties, we are going to construct a sample page to extract data. The aim to include this chapter is to reflect which the ideal Web page that gives facilities to our tools to extract data is.

As easy to imagine, this sample page is going to be constructed avoiding all the previous commented problems. It will have the following characteristics:

- Structured data representation
- HTML code following the W3C standard
- No nested data elements
- Structure containing the same type of elements
- Used Flash or scripts don't contain data to be extracted
- Use of CSS Styles to identify and give format to elements

Taking a random scenario for this Web page, we have built a Top10 page from the users that gained more points in a strategy game. The next screenshot give us an accurate idea on how does it look like:



Top 10 Users Classification

Kings of Sun 2008 Contest

This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each year online competitions take place. The user has to use his strategy habilities to be the best king of his land, that includes have a growing population, construct temples, study new technologies, begin wars to extend territory...

User	Military points	Technology Points	Religion Points	Social Points	Total Points
Player 1	359	566	45	411	1381
Player 2	320	468	69	402	1259
Player 3	322	365	78	325	1090
Player 4	302	375	45	344	1066
Player 5	267	333	43	377	1020
Player 6	233	340	68	258	879
Player 7	252	340	38	238	868
Player 8	214	268	47	222	751
Player 9	196	280	44	223	743
Player 10	188	199	22	123	532

Figure 7: Screenshot of the used scenario

The structure of the data follows a logic order. From top it is shown the following elements; the main title and the second title, a short description of the contest, the result table, a banner in Flash and a link to the main menu of the page. By observing this structure we conclude that there are no elements mixed, we mean for example that a second part of the description is not placed after the result table or that the flash banner is not located between some of the rows.

We have designed this HTML code accomplishing the W3C standard and as it is visible no nested elements appear. This page is static too, so it means that the structure of the content is not going to change.

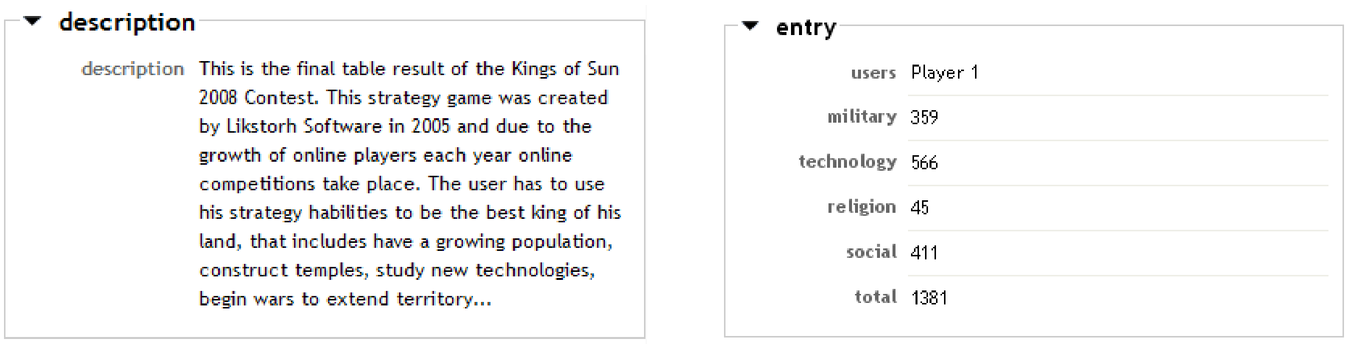
If things were more complicated and we decide to realize a query to a database getting the 10 users with higher points and use PHP to write these results to the table, no problems will appear. This happens because we are only doing modifications to the data, the structure and all the other characteristics of our HTML source remain without changes.

We have inserted dynamic content too, specifically a banner in Flash, in this case we don't care about this as it doesn't contain important data to extract.

On the other hand CSS styles have been used to give format to the different types of content. This will help our tools to separate the content by using the class attribute from DIV or SPAN tags.

```
<style type="text/css">
<!--
body {
    background-image:  url(stripe_0f0f5a1a754d31e5d683ef68f17e2986.png);
}
.title {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 24px;
    color: #FFFFFF;
}
.subtitle {font-family: Verdana, Arial, Helvetica, sans-serif}
.description {font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 12px; }
.table_data {font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 12px; }
.link {
    color: #FFFFFF;
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 12px;
}
-->
</style>
```

After describing all the characteristics of the page we have realized an extraction using one of our testing data extraction tools and all the data has been extracted correctly.



description

description This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each year online competitions take place. The user has to use his strategy habilities to be the best king of his land, that includes have a growing population, construct temples, study new technologies, begin wars to extend territory...

entry

users	Player 1
military	359
technology	566
religion	45
social	411
total	1381

Figure 8: Some results of the data extraction

3- Data extraction tools

In this chapter we are going to present all the information referred to the Web data extraction tools. First of all we are going to describe shortly all of their characteristics.

In the end, a brief tool comparative is presented to directly compare the features of all of them.

3.1- Related work

The fact of creating the group of tools used to perform extractions it is an important decision in our work. Selecting an specific group of tools with some characteristics or another one with others could lead to different results.

We decided to work with a group of ten tools. The methodology followed to realize the selection is focused on searching into papers and related documents of Web data extraction tools, realize searches through Google and through Sourceforge.

A good deal of information that helped us to realize the selection has been extracted from papers. Especially we used a paper named *A brief Survey of Web Data Extraction Tools* [24]. This paper contains a categorization of the data extraction tools and explains the characteristics of each group.

Searches into Google don't have always contributed to find a suitable tool. On the other hand by Sourceforge we could find several useful open-source tools but sometimes unfinished or not suitable projects.

In fact we decided to work with HTML-aware tools. These kinds of tools, which features are going to be explained in the next section, are characteristic for its level of automation. We don't have to spend a big amount of time in the configuration process to realize extractions but on the other hand most of them can only extract information from HTML files. However we are more interested in the automation degree than to the source structure of our data, as we are going to focus our work on HTML extractions. In this way, we can say that data extractions normally work with big amounts of data and this kind of tools are designed to automate this process.

We realized an heterogeneous selection of these tools, which means that we have considered a variety of characteristics between them: commercial and non commercial tools, GUI and non GUI support tools, Linux and Windows tools… The aim of this variety consist of having a general sample set of tools and see if their different characteristics affect to the data extraction features. Other possible groups could be formed with similar characteristics but from a global view our selection let us reach our goal to realize a final categorization.

We can find all these 10 tools in the section overview of the tools. A brief description of each tool with his main characteristics is shown. Next we are going to introduce a taxonomy to characterize them and give the reader a general eyesight.

3.2- A taxonomy for characterizing Web data extraction tools

This general presented taxonomy is based on the main technique used by each tool to generate a wrapper, what led us to the following group of tools: Languages for Wrapper Development, HTML-aware Tools, NLP-based Tools, Modeling-based Tools and Ontology-based Tools.

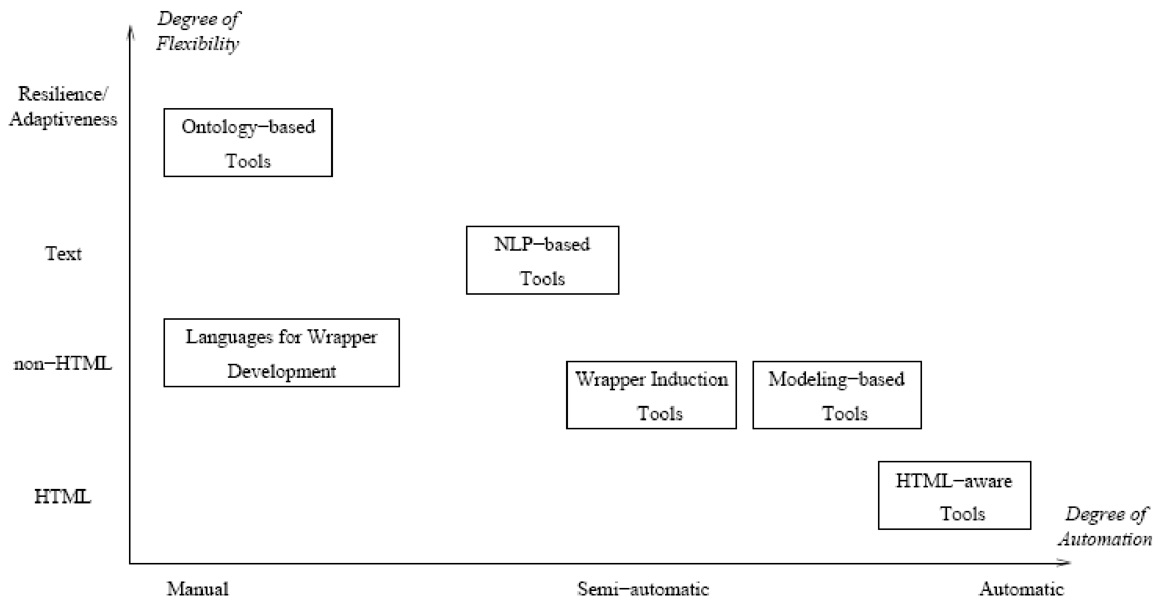


Figure 9: Classification using the flexibility and automation degree of Web data extraction tools – from [24]

As shown on Figure 9, a classification using the flexibility degree and the automation degree could be constructed. Generally, the more automated a tool is, the less flexibility the degree has. We can find different grades of flexibility; from treating standard HTML documents to those ones having strong resilience/adaptiveness properties. There are also several grades of automation that vary from manual to automatic.

Ontology-based tools are the ones that have the best flexibility but at the same time they have to be configured in a manual way. We are going to concentrate our effort in HTML-aware tools, that have a high degree of automation but they could only extract information from HTML.

Explained in a brief way these are the main characteristics of each type of tools found in the taxonomy:

- Languages for Wrapper Development: It was one of the first initiatives to assist users in constructing wrappers. These languages were proposed as alternatives to general purpose languages such as Perl or Java, which were prevalent so far for this task. Some of the best known tools that adopt this approach are Minerva, TSIMMIS, Web-OQL.

- Ontology-based: This type of tools relies directly to the data to perform data extractions. Given a specific domain application, an ontology can be used to locate constants present in the page and to construct objects with them. The most representative tool of this approach is BYU.

- NLP-based: This type of tools uses Natural language processing (NLP) techniques to learn extraction rules for extracting relevant data existing in natural language documents. They use rules based on syntactic and semantic constraints that help to identify relevant information within a document. The most representative tools of this approach are RAPIER, SRV and WHISK.

- Wrapper induction: They generate delimiter-based extraction rules derived from a given set of training examples. The main distinction between these tools and those based on NLP is that they don't rely on linguistic constraints, but rather in formatting features that implicitly delineate the structure of the pieces of the data found. Some tools of this approach are WIEN, SoftMealy and STALKER.

- Modeling-based: They are based in the fact that given a target structure for objects of interest, they try to locate in Web pages portions of data that implicitly conform to that structure. The structure is provided according to a set of modeling primitives (e.g, tuples, lists, etc.) that conform to an underling data model. Tools that adopt this approach are NoDoSE and DEByE.

- HTML-aware: This type of tools relies on inherent structural features of HTML documents for accomplishing data extraction. A transformation of the source document to a parsing tree is realized and it reflects its HTML tag hierarchy. Therefore, extraction rules are generated either semi-automatically or automatically and then applied to the tree. In these documents we are going to use tools that follow this approach, some of them are RoadRunner, XWRAP or Robomaker.

In the following sections we are going to explain the main characteristics of the set of HTML-aware tools that we have selected.

3.3- Overview of tools

The aim of this section is to give a general view to the reader of each of the tools used in this document. Its main features are shown here and in almost all of them a screenshot is presented.

- **Dapper**

Dapper is an online tool which allows the user to extract information from Websites. To use it all what we need is an Internet browser and Internet connection as this service is only available online. Dapper is at the moment in beta phase but it is totally functional.

The usage of Dapper is totally free and we only need to create a new account to use it. We can create our own wrappers (or Dapps as they are called) or use wrappers already created from other registered users.

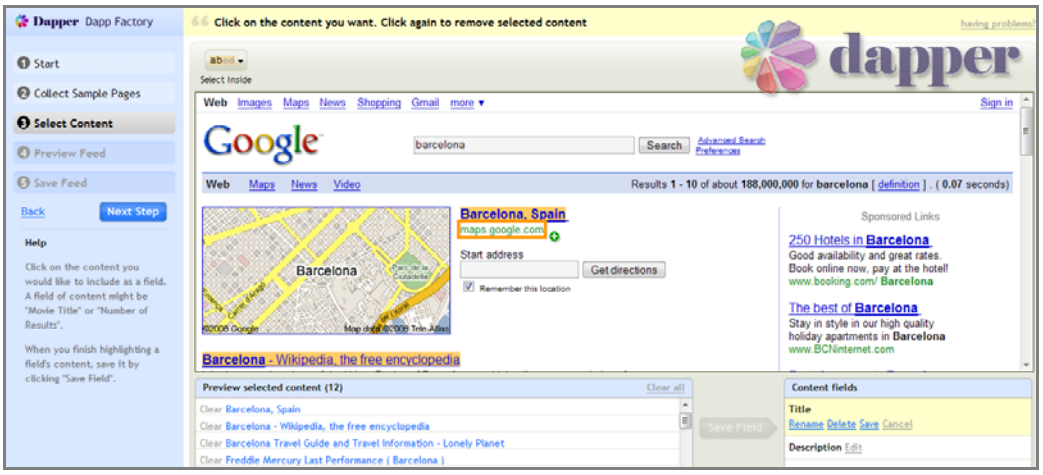


Figure 10: Dapper Screenshot

Dapper is one of the easiest tools to use as its interface is totally graphical. Apart from extracting data it allows you to create Flash widgets or alerts using the extracted information. Link a Dapp output to another Dapp input to create some new Dapps is another useful functionality.

▪ **Robomaker**

Robomaker is a Web 2.0 developer platform for creating mashups. The tool lets the user create RSS feeds, REST Web Services or Webclips in few steps.

It is provided with powerful programming features including interactive visual programming, full debugging capabilities, an overview of the program state and easy access to context-sensitive online help, this features make it really complete and dynamic. It can be used in both Windows and Linux platforms.

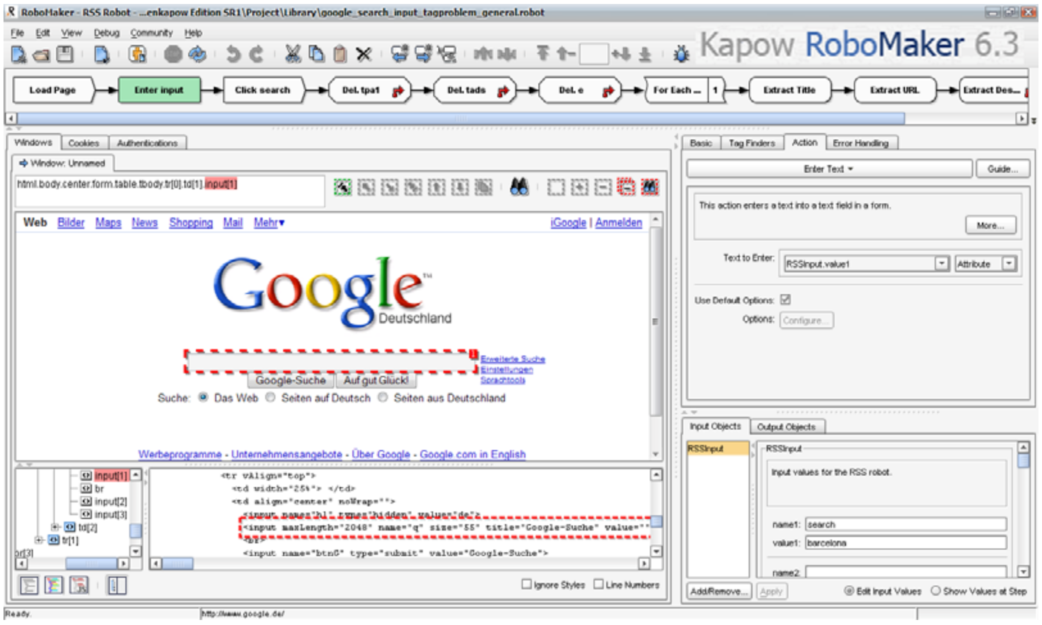


Figure 11: Robomaker Screenshot

▪ **Lixto**

The Lixto Visual Developer (VD) is a software tool that allows the user to define wrappers, which visually access data in a structured way, as well as configuring the necessary Web connectors.

The program is originally from a research project of the Technical University of Vienna that becomes later in the Lixto Software. It provides businesses with effective, user-friendly, and time critically viable wrapping, integration and delivery of information all in the same product.

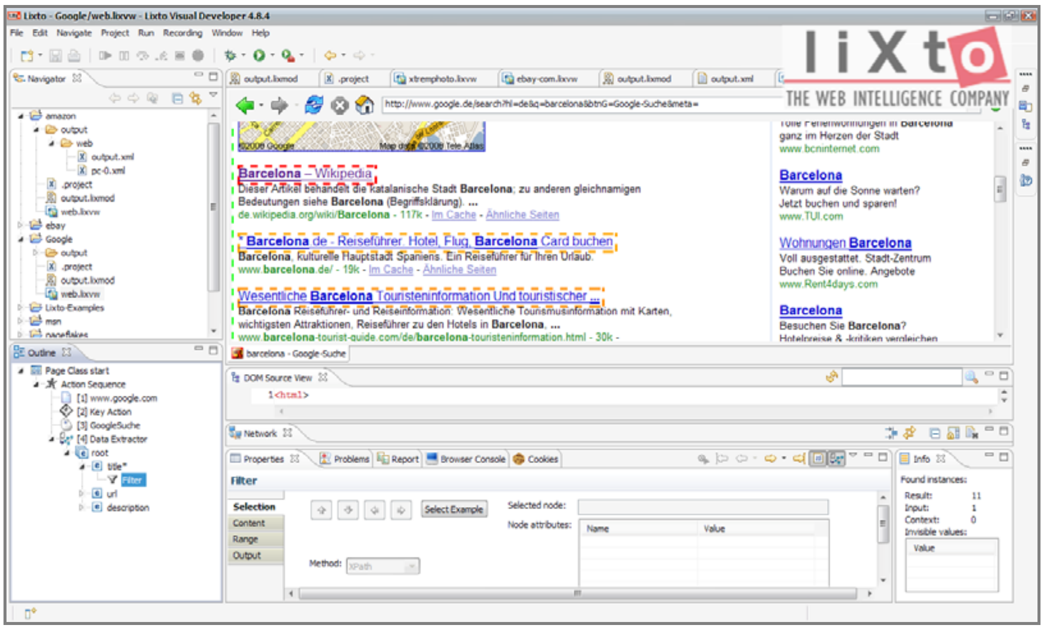
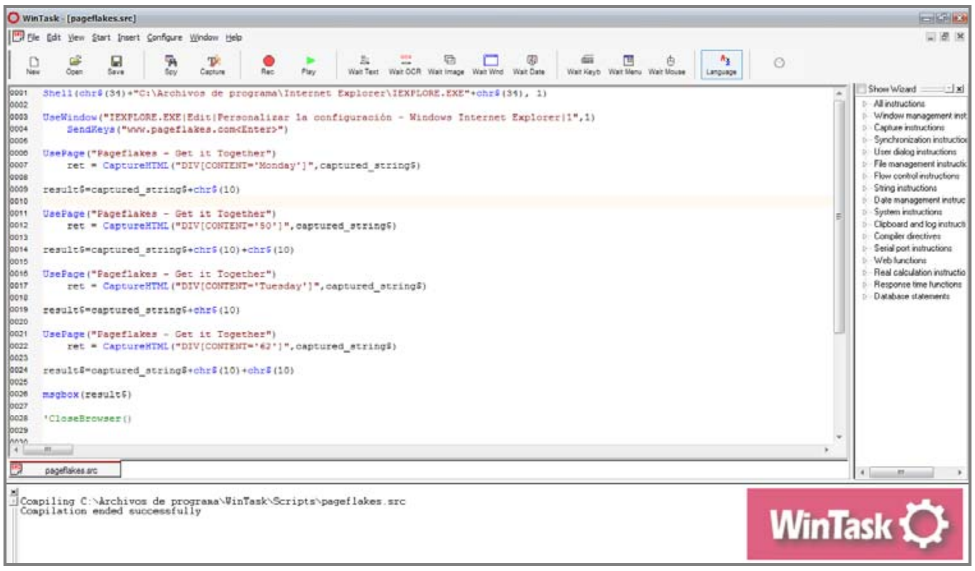


Figure 12: Lixto VD Screenshot

▪ **WinTask**

WinTask is a Windows tool used to automate repetitive tasks or actions which should run at a certain moment. One of its features is data extraction of Web sites.

WinTask can launch the URL to load, send a userid and an encrypted password if it is a secure site, conduct searches, and navigate to the different pages where some field contents have to be extracted. This tool is only available in the trial-version, if we want full functionality we have to buy it. It works by using its own scripts so at the beginning it can be a little hard to familiarize with the syntax.



```
Shell(chr$(34)+"C:\Archivos de programa\Internet Explorer\IEXPLORE.EXE"+chr$(34), 1)

UseWindow("IEXPLORE.EXE|Edit|Personalizar la configuración - Windows Internet Explorer|1",1)
     SendKeys("www.pageflakes.com<Enter>")

UsePage("Pageflakes - Get it Together")
     ret = CaptureHTML("DIV[CONTENT='Monday']",captured_string$)

result$=captured_string$+chr$(10)

UsePage("Pageflakes - Get it Together")
     ret = CaptureHTML("DIV[CONTENT='50']",captured_string$)

result$=captured_string$+chr$(10)+chr$(10)

UsePage("Pageflakes - Get it Together")
     ret = CaptureHTML("DIV[CONTENT='Tuesday']",captured_string$)

result$=captured_string$+chr$(10)

UsePage("Pageflakes - Get it Together")
     ret = CaptureHTML("DIV[CONTENT='62']",captured_string$)

result$=captured_string$+chr$(10)+chr$(10)

msgbox(result$)

'CloseBrowser()
```

```
Compiling C:\Archivos de programa\WinTask\Scripts\pageflakes.src
Compilation ended successfully
```

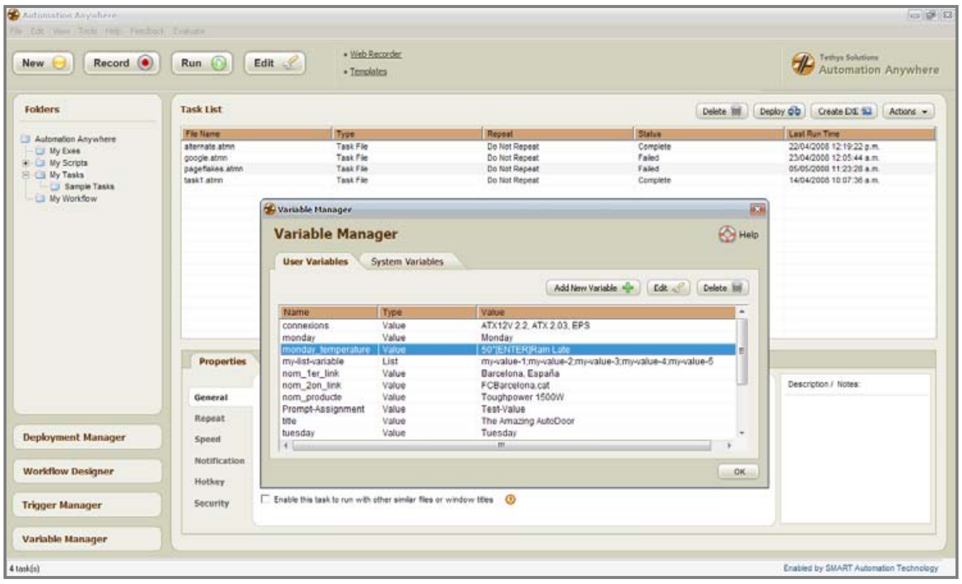
WinTask - [pageflakes.src]
File Edit View Start Insert Configure Window Help
New Open Save Spy Capture Rec Play Wait Text Wait OCR Wait Image Wait Wnd Wait Date Wait Keyb Wait Menu Wait Mouse Language
pageflakes.src
WinTask

Figure 13: WinTask Screenshot

- **Automation Anywhere**

Automation Anywhere is a Windows tool that lets the user record click and mouse movements and to create tasks in desktop that could interact with our programs. It can also record from the Web, this consists basically of creating a navigation sequence and extract data of our interest.

We can also use templates to realize concrete tasks or use the task editor that lets the user create a task using some predefined actions, conditions, scripts, mouse and keyboard activity... This tool is only available in the trial-version, if we want full functionality we have to buy it.



Task List

File Name	Type	Repeat	Status	Last Run Time
alternate.atmn	Task File	Do Not Repeat	Complete	22/04/2008 12:19:22 p.m.
google.atmn	Task File	Do Not Repeat	Failed	23/04/2008 12:05:44 a.m.
pageflakes.atmn	Task File	Do Not Repeat	Failed	05/05/2008 11:23:28 a.m.
task1.atmn	Task File	Do Not Repeat	Complete	14/04/2008 10:07:36 a.m.

Variable Manager

Name	Type	Value
connexions	Value	ATX12V 2.2, ATX 2.03, EPS
monday	Value	Monday
monday_temperature	Value	50°[ENTER]Rain Late
my-list-variable	List	my-value-1;my-value-2;my-value-3;my-value-4;my-value-5
nom_1er_link	Value	Barcelona, España
nom_2on_link	Value	FCBarcelona.cat
nom_producte	Value	Toughpower 1500W
Prompt-Assignment	Value	Test-Value
title	Value	The Amazing AutoDoor
tuesday	Value	Tuesday

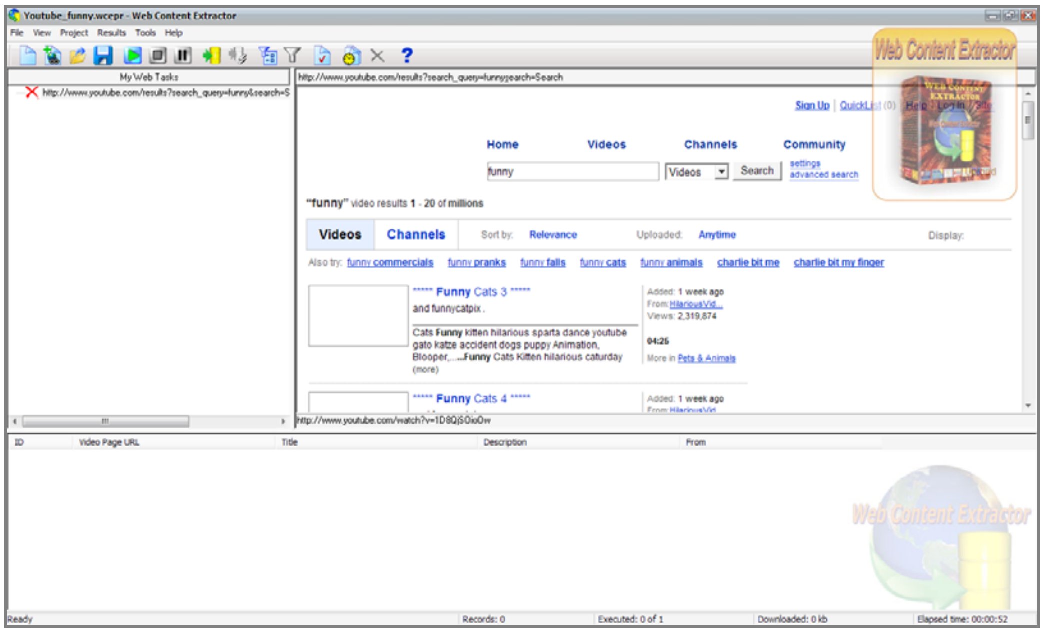
Automation Anywhere
New Record Run Edit
Web Recorder
Templates
Tethys Solutions Automation Anywhere
Folders
Automation Anywhere
My Exes
My Scripts
My Tasks
Sample Tasks
My Workflow
Delete Deploy Create EXE Actions
User Variables System Variables
Add New Variable Edit Delete
Help
OK
Enable this task to run with other similar files or window titles
Properties
General Repeat Speed Notification Hotkey Security
Description / Notes:
Deployment Manager
Workflow Designer
Trigger Manager
Variable Manager
4 task(s)
Enabled by SMART Automation Technology

Figure 14: Automation Anywhere Screenshot

▪ Web Content Extractor

Web Content Extractor is a Windows tool that allows the user to create a project for a particular site, extract data from it and store it in the current projects database. The extracted data can be exported to a variety of formats including Microsoft Excel (CSV), Access, TXT, HTML, XML, SQL script or MySQL script.

As it happens with the two tools analyzed before, we could only download the trial-version of Web Content Extractor.



Youtube_funny.wcepr - Web Content Extractor
File View Project Results Tools Help
My Web Tasks
http://www.youtube.com/results?search_query=funny&search=Search
Home Videos Channels Community
"funny" video results 1 - 20 of millions
Videos Channels
ID Video Page URL Title Description From
Ready Records: 0 Executed: 0 of 1 Downloaded: 0 kb Elapsed time: 00:00:52

Figure 15: Web Content Extractor Screenshot

▪ Roadrunner

Roadrunner is a project of the database departments of the *Università di Roma Tre* and the *Università della Basilicata*. This tool generates a wrapper for the analysis of similarities and differences from several sample files of the same class.

With this tool, a class is an amount of pages generated by the same script, so structurally the same, but in some places both content are quantitatively different. This wrapper is a representation of the investigated sample files in the form of a regular expression or so-called union-free regular expression (UFRE).

▪ XWRAP

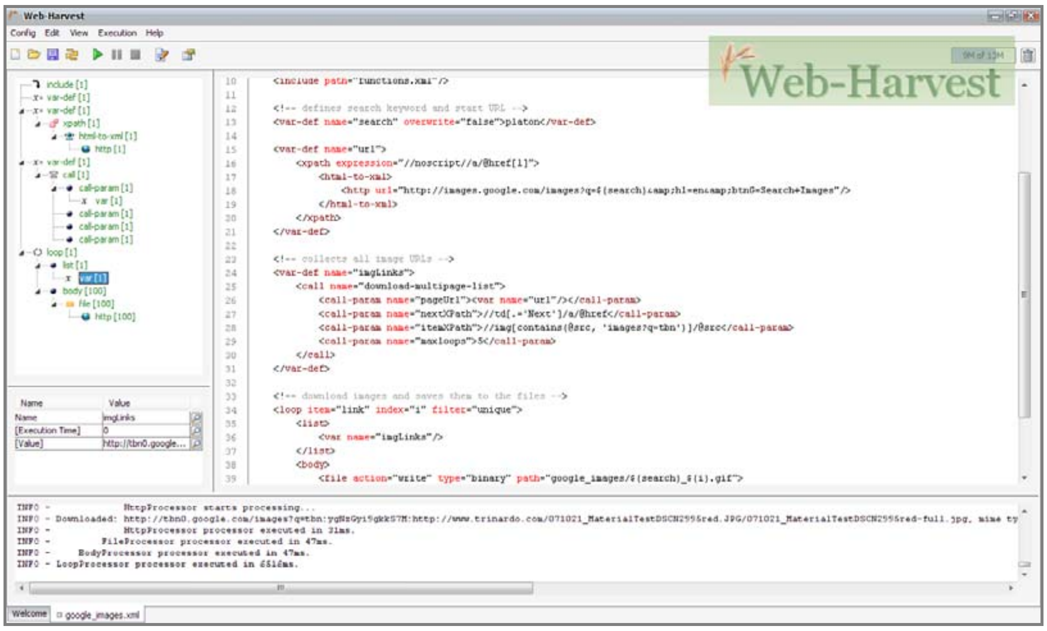
XWRAP is a tool that was developed at the Georgia Institute of Technology. Its developers described it as an XML-enabled wrapper construction system for Web information sources.

The toolkit includes three components: Object and Element extraction, filter interface extraction and code generation. The wrappers are generated as Java classes. To use it we have to enter the URL of our desired Web site and the customization of the extraction process results is done via the Web by XWRAP.

To use XWRAP we need a separate Web server (such as Apache Tomcat).

- **Webharvest**

Webharvest is an Open Source Web Data Extraction tool written in Java. It offers a way to collect desired Web pages and extract useful data from them. In order to do that, it leverages well established techniques and technologies for text/XML manipulation such as *XSLT*, *XQuery* and *Regular Expressions*. Web-Harvest mainly focuses on HTML/XML based Web sites which still make vast majority of the Web content.



```
<include path="functions.xml"/>

<!-- defines search keyword and start URL -->
<var-def name="search" overwrite="false">platon</var-def>

<var-def name="url">
    <xpath expression="//noscript//a/@href[1]">
        <html-to-xml>
            <http url="http://images.google.com/images?q=${search}&amp;hl=en&amp;btnG=Search+Images"/>
        </html-to-xml>
    </xpath>
</var-def>

<!-- collects all image URLs -->
<var-def name="imgLinks">
    <call name="download-multipage-list">
        <call-param name="pageUrl"><var name="url"/></call-param>
        <call-param name="nextXPath">//td[.='Next']/a/@href</call-param>
        <call-param name="itemXPath">//img[contains(@src, 'images?q=tbn')]/@src</call-param>
        <call-param name="maxloops">5</call-param>
    </call>
</var-def>

<!-- download images and saves them to the files -->
<loop item="link" index="i" filter="unique">
    <list>
        <var name="imgLinks"/>
    </list>
    <body>
        <file action="write" type="binary" path="google_images/${search}_${i}.gif">
```

Name	Value
Name	imgLinks
[Execution Time]	0
[Value]	http://tbn0.google...

```
INFO -             HttpProcessor starts processing...
INFO - Downloaded: http://tbn0.google.com/images?q=tbn:ygNzGyi5gkkS7M:http://www.trinardo.com/071021_MaterialTestDSCN2995red.JPG/071021_MaterialTestDSCN2995red-full.jpg, mime ty
INFO -             HttpProcessor processor executed in 31ms.
INFO -         FileProcessor processor executed in 47ms.
INFO -     BodyProcessor processor executed in 47ms.
INFO - LoopProcessor processor executed in 6516ms.
```

Figure 16: Web-Harvest Screenshot

- **Goldseeker**

Goldseeker is a data extraction tool, specifically a script under the GNU LGPL license. It was built to extract formatted data from HTML files, but it can be used with all kind of files. Its behavior is defined by a rule-based configuration file. It can process files on the local server or directly get Web pages via Internet. It is a development version, uncommented, undebugged and unfinished. Nevertheless, it can already be used for simple extractions.

3.4- Descriptive comparison of HTML-based tools

Once each of the tools has been introduced, we are going to present some of its characteristics. Two tables have been realized to show, in the first one, a basic overview of them and, in the second one, the data extraction tools features. A categorization of the tools using distinguishing features is presented too.

	Usage	Installation	Interface	Extraction	Free
Dapper	Online	Execution	Internet browser	Allow input variables, several formats, easy to use	Yes
Robomaker	Online	Windows installation	Program GUI, Internet browser	Allow input variables, complete functionality, several formats, medium complexity	Yes
RoadRunner	Local	Linux installation, configuration	Linux Shell	Use of configuration files, working with static content, complex to use	YES, GNU GPL License
XWRAP	Online + Tomcat	Configuration	Internet browser	Medium complexity, working with static content	Yes
Lixto	Online	Windows installation	Program GUI, Internet browser	Allow input variables, Scripts usage, medium complexity, Web recording tool	No, requires license
WebHarvest	Online	Execution	Program GUI	Medium complexity	Yes
GoldSeeker	Local or Online (PHP support)	Configuration	Internet browser	Scripts usage, in development, simple extraction uses	Yes, GNU LGPL License
WinTask	Local, Online	Windows Installation	Program GUI, Internet browser	Scripts usage, Working with static content, Web recording tool	No
Automation Anywhere	Local, Online	Windows Installation	Program GUI, Internet browser	Output to program variables, Web recording tool	No
Web Content Extractor	Local, Online	Windows Installation	Program GUI, Internet browser	Working with static content, several formats	No

Figure 17: Data extraction tools overview

A first categorization of the tools has been realized selecting distinguishing features to organize groups containing a set of tools with the same characteristics. We have used a tree structure to represent this categorization as it reflects the result in a visual and clear way, it is shown in the following figure:

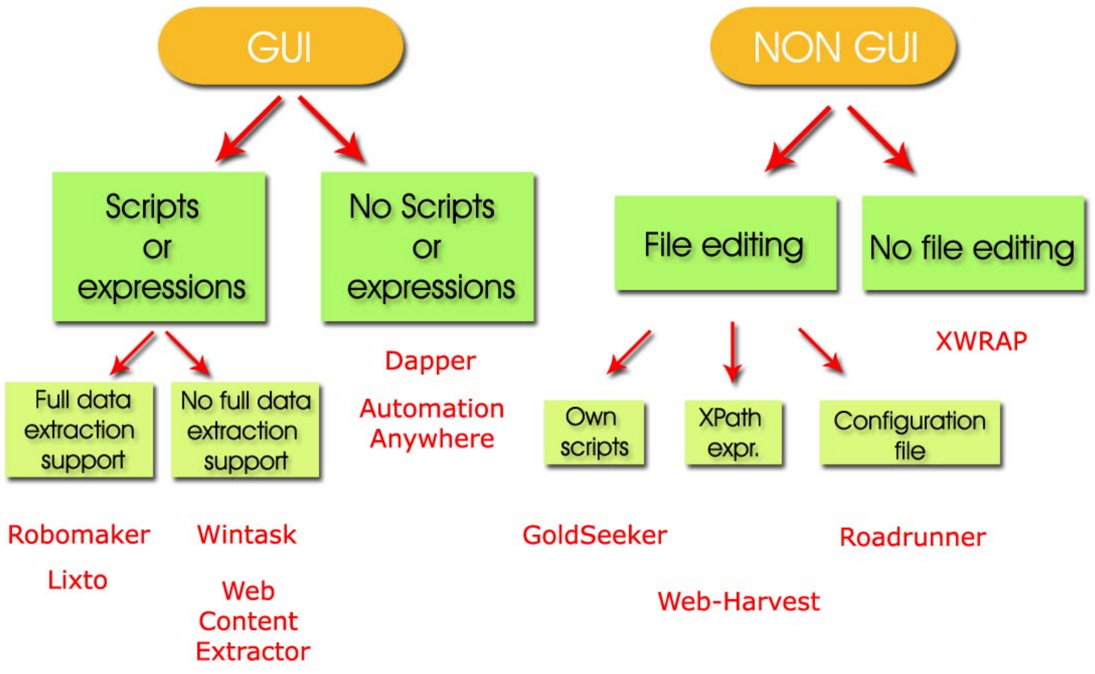


Figure 18: Tools categorization using distinguishing features

In the first level of the tree we can split the tools into two groups by making a distinction among the GUI. This characteristic directly let us to have a fully distinguished group of tools, the one that has GUI and the other one that doesn't. The fact of having a GUI makes it easier for the user. He has more options and menus to interact and a real time visualization of the elements that are being selected to extract information.

When speaking about the GUI tools, the next distinguishing feature consists of scripts and expressions. This feature makes a tool more powerful and lets extract data in a more precise way, so that is a really important point to take into account. Once realized a group that use expressions or scripts, the next characteristic to make a new separation is the full support to the data extraction.

On the other hand when speaking about the non GUI tools, we can realize a main separation by the necessity to edit a configuration file to prepare the data extraction process. When the property is true a new separation could be realized taking care of the configured file type.

This tree representation is useful to construct groups taking into account structural characteristics of our tools.

	EXTRACTION FEATURES								
	Input variables	Scripts Usage	Output Formats	Complexity	Non static content pages	Extract contents from more than one Page	Error treatment	Execution Time	HTML or other documents
Dapper	Yes	No	XML, RSS, HTML, Google Gadget, Netvibes Module, PageFlake, Google Maps, Image Loop, Icalendar, Atom Feed, CSV, JSON, XSL, YAML, email	Low	Yes	No	No	Very Good	HTML
Robomaker	Yes	Yes, Javascript	RSS/Atom Feed, REST Web Service, Web Clip	Medium	Yes	Yes	Yes	Very Good	HTML
RoadRunner	No	No	XML, HTML	High	No	No	No	Good	HTML
XWRAP	No	No	Java	Medium	Yes	No	No	Good	HTML
Lixto	Yes	Yes	XML	Medium	Yes	Yes	No	Very Good	HTML
WebHarvest	No	No	XML	Medium	Yes	No	No	Good	HTML
GoldSeeker	No	Yes	Text	Medium	Yes	No	No	Poor	HTML and documents
WinTask	By script	Yes	File, Excel, DB	Medium	No	Yes	No	Good	HTML and documents
Automation Anywhere	No	No	File, Excel, DB, EXE	Low	No	Yes	No	Good	HTML and documents
Web Content Extractor	No	No	File, Excel, DB, SQL script File, MySQL script File, HTML, XML, HTTP submit	Low	No	No	No	Poor	HTML

Figure 19: Data extraction tools features

This table has been created selecting a set of different characteristics that can be evaluated in all of our tools. The aim of this table is to give to the user a general view of them, making a first comparison, and to have an idea of the main differences that exist.

The field input variables is useful to introduce information in form of fields, the user could need this feature when expecting results from a non static Web page. Through scripts the tools have a powerful way to threat with the information and the property of working with non static content pages makes the tool able to work in a bigger number of actual Web pages having dynamic content. Another important feature is to know if our tool is able to extract information from more than one Web page at the same time, this is useful to join more than one normal page extraction.

General features that can be found in most of the programs can be found in the data extraction tools. Fields like complexity, error treatment, execution time or input and output formats.

Next the filled fields from the data extraction tools are commented in detail:

- **Input variables:** That field refers if we can use an input variable to use in form fields to get dynamic results. Changing the value of this variable we can obtain new results. This is really useful when performing searches for example using Web search engines.

  Values: Yes / No

- **Scripts usage:** Usage of scripts gives the tools more flexibility to interact with the extracted data and to perform transformations. Sometimes it is hard to familiarize with the syntax but upon learnt they are useful to perform complex tasks that are difficult to realize in a visual way.

  Values: Yes / No

- **Output formats:** This is the list of the output formats that the tool can export.

- **Complexity:** It measures the complexity when using our tool to perform extractions.

  Values: Low / Medium / High

- **Non static content pages:** This field refers if the tool is suitable to extract data from pages that are applicant to perform changes on its content. For example the result search pages from Web search engines.

  Values: Yes / No

- **More than one page:** It refers if we can get data from more than one page at the same time. Useful for example in several search results from Web search engines.

  Values: Yes / No

- **Error treatment:** This field refers if the application has a way to treat errors when performing data extractions.

  Values: Yes / No

- **Execution time:** This field refers to how much time requires a tool to perform data extractions.

  Values: Very Poor, Poor, Good, Very Good

- **HTML or other documents:** It refers if we can only extract data from HTML sources or others.

  Values: HTML, documents

4- Tests using the data extraction tools

To evaluate the quality of the extraction tools a set of tests have been developed. Our goal is to see the behavior of these tools and if they could extract the data that we expect.

4.1- Overview of tests

To realize these tests we have thought about using known and often visited Web Pages where an enterprise or a single user can found interest on extracting data.

This set of tests tries to embrace several aspects of the data extraction process considering several situations with specific characteristics. A general view of them presenting the used sources and the goals of each one is shown in the next table:

Test	Used source	Goal
General tests	Kings of Sun 2008 Contest (self built HTML source), Google, Yahoo! Search, MS Live Search, Ebay, Pageflakes	Try to extract data using Web data extraction tools from general Web Pages. The main goal is to extract data from actual and known Web pages taking care of analyzing several of the presented features of the section 3.4
Resilience tests against changing HTML code	Amazon	Try to see how robust our tools are against changes to the HTML code. As it is one of the most important problems of the Web data extraction we have dedicated several tests modifying a sample page of Amazon.com
Precision tests of the extracted data	List of published books (self built HTML source)	Try to see which the precision of our tools when talking about extraction of concrete fields of data is. This feature is really important when we consider that we want to extract concrete information and not only an entire field of data (i.e. author of an article, date of publication…)

Figure 20: General overview of used tests

The question that answers why we have selected a set of tests to realize the extractions is because we need a process to qualify the extraction of each of the tools, analyzing several of the features presented in the section 3.4 and to get tangible results to elaborate the final categorization of the tools.

Other types of tests could be used to achieve similar results but somehow the ones selected take a global view of the extraction features and are suitable to be used.

The general tests embrace several scenarios, from basic data extraction to dynamic content pages. They are representative and give an idea of the behavior of our tools in several situations.

Resilience and precision tests have been introduced to evaluate these two particular features, both really important when extracting Web data.

4.2- Methodology

To realize these tests a methodology has been elaborated to get final results and to know which the steps are that we are going to follow. The next figure illustrates the used methodology used when performing our data extraction tests:

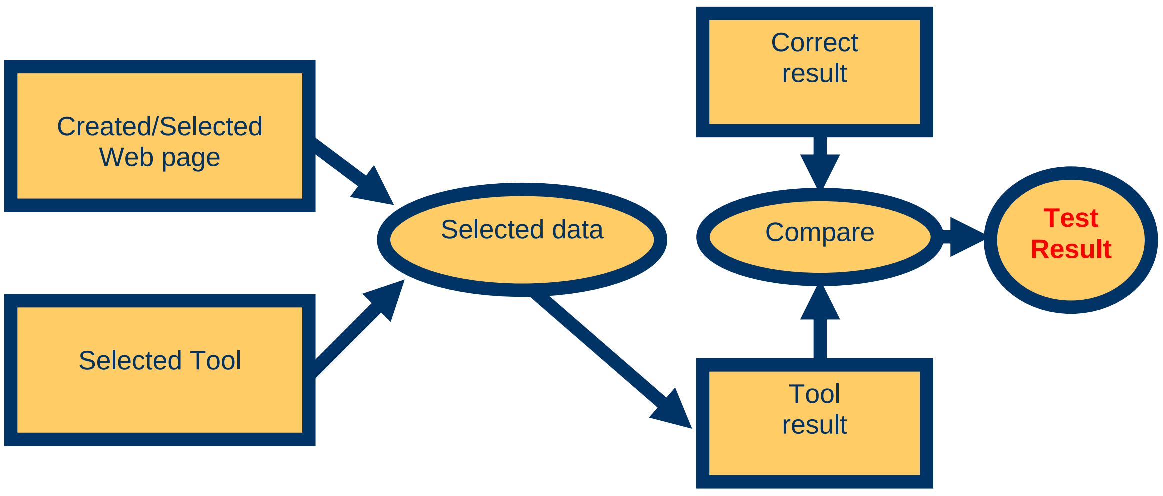


Figure 21: Used methodology for the data extraction tests

The first step consists of creating or selecting a Web page source in which we want to extract data. After that, we select the data extraction tool with we are going to perform the test.

Most of the selected Web page sources can be found on the Web- However, self-made Web pages have been created to focus in some of the features that we want to test. To elaborate these self made sources we have used *Adobe Macromedia Dreamweaver* to create and edit the content together with a private Web server to locate the files. To upload the data files we have used an FTP client.

Next we configure our tool to extract the data, this process varies depending the selected tool. Then we receive an output from this tool and the resulting extracted data is compared with the correct extracted data.

This comparison allows qualifying the data extraction results of the analyzed tool. Several degrees of qualification have been used: for example a poor, good or very good data extraction. We can also give an explanation of why the data has not been extracted correctly. These are possible ways to realize a conclusion of the test.

Once we get all the final results a conclusion table is presented with a summary of all of them and a general view to the reader can be presented. These conclusion results are indeed used too to elaborate the final categorization section.

4.3- Problems with some of our tools

Before starting, we have to mention that we have experienced some problems with XWRAP and Roadrunner.

As explained in the tools introduction chapter, XWRAP is a tool developed at the Georgia Institute of Technology. It has been installed without problems in our computer, we have configured the tool to realize data extractions from our scenario, and we got a resulting Java file to execute the wrapper.

As it states in the Web of the XWRAP project, there exists only three ways to realize a data extraction:

- Register the wrapper to the GT Wrapper Repository
- Download the wrapper package and integrate it into our own Java program
- Download the wrapper package and run the wrapper on the command line

Unfortunately, we could not realize a satisfactory execution of one of these options. Neither the GT wrapper Repository was available through the XWRAP Elite Home Page and nor the link to download the wrapper package to execute the wrappers.

In conclusion we could configure the Web data extractions but we could not retrieve any final results due to the unavailability of sources from the XWRAP Elite Home Page. The last update of this Web page was on April 2000. Up to now no more updates have been carried out.

Due to this fact, we could not get self conclusions of the quality of the extractions performed by XWRAP and we are going to exclude this tool from our tests.

We could realize extractions and achieve results using Roadrunner. This tool infers a grammar for the HTML code to generate a wrapper for a set of HTML pages and then uses this grammar to parse the page and extract pieces of data. That is to say it doesn’t rely on user-specified examples and does not require any interaction with the user during the wrapper generation process. This means that wrappers are generated and data is extracted in a completely automatic way.

The system works with two HTML pages at a time and pattern discovery is based on the study of similarities and dissimilarities between the pages.

The tests presented in the following section are thought to extract concrete data of a set of HTML page sources, the same occurs when evaluating the resilience and precision properties. Due the way to proceed of this tool we can not achieve self conclusions using our tests, and then, we are going to exclude it.

4.3- General data extraction tests

In this section, we are going to use all the extraction tools to perform several general tests. The aim of this section is to test most of the general features of our data extraction tools. In each performed test is explained which features we are going to test.

4.3.1- Basic data extractions

In this section we are going to extract information from a simple Web page. This means we are not going to realize extractions that have the necessity of specific features to perform an extraction. The target to introduce these basic tests is to see if our tools can extract data from basic Web sources.

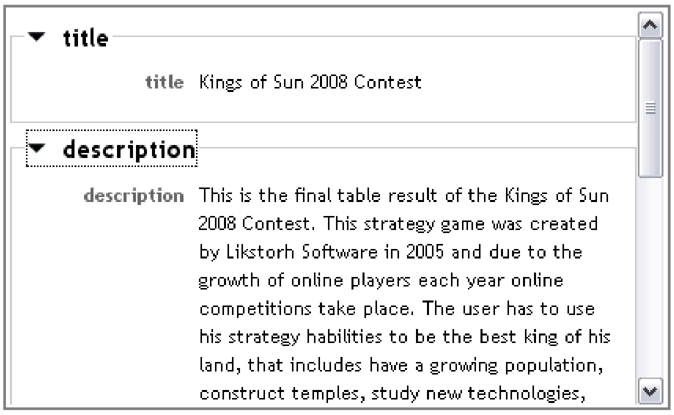
With this kind of test we want to include all the usual extractions that can be found in a normal HTML file. It doesn't matter if the data is retrieved direct from an URL or from a file.

We have used a previous scenario to realize the tests, it is the Kings of Sun 2008 Contest of the chapter 2.4. We selected this scenario as it has a basic HTML structure and it will make the things easier and clearer when presenting the results.

From all the information found on this page we are going to extract the title, the short description and the list of the player names. By extracting such fields we can realize a conclusion of basic HTML extractions.

■ **Dapper:**

With Dapper after following the standard steps to select the content of interest we could receive all the information without problems. We grouped the information distinguishing the main title, the description and the players list.

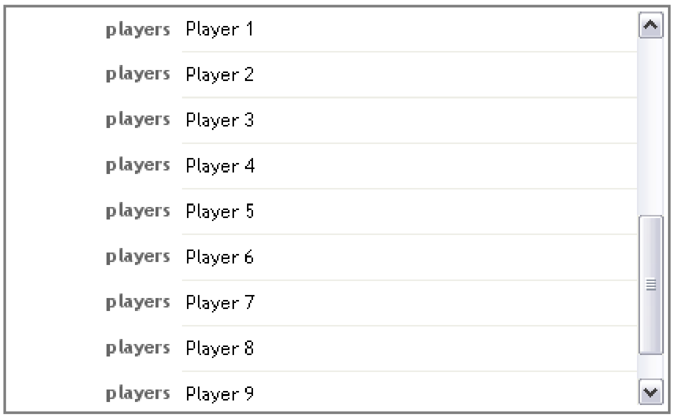


▼ title

title Kings of Sun 2008 Contest

▼ description

description This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each year online competitions take place. The user has to use his strategy habilities to be the best king of his land, that includes have a growing population, construct temples, study new technologies,

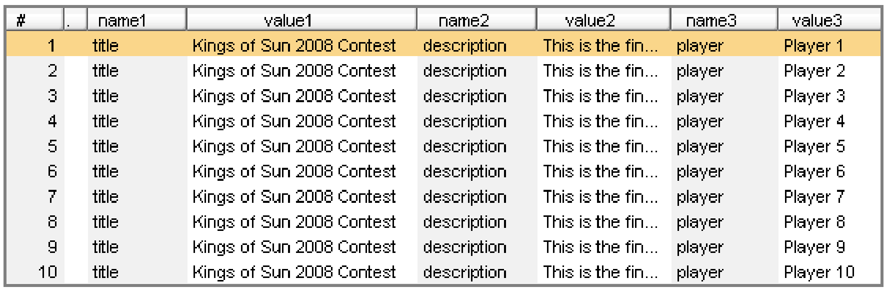


players	Player 1
players	Player 2
players	Player 3
players	Player 4
players	Player 5
players	Player 6
players	Player 7
players	Player 8
players	Player 9

Figure 22: Final output using Dapper

■ Robomaker:

Robomaker presented no problems when extracting simple data. We only had to select the title and the description to extract these fields and introduce a loop to select all the players. Following are the final results:



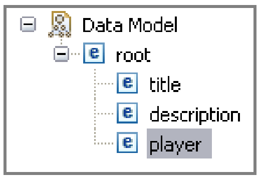
#	.	name1	value1	name2	value2	name3	value3
1		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 1
2		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 2
3		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 3
4		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 4
5		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 5
6		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 6
7		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 7
8		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 8
9		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 9
10		title	Kings of Sun 2008 Contest	description	This is the fin...	player	Player 10

Figure 23: Final output using Robomaker

■ Lixto:

First of all we have to know that Lixto VD only extracts our results in the XML format. First of all we have to create a Lixto Data Model to specify how the output to our XML file will be.

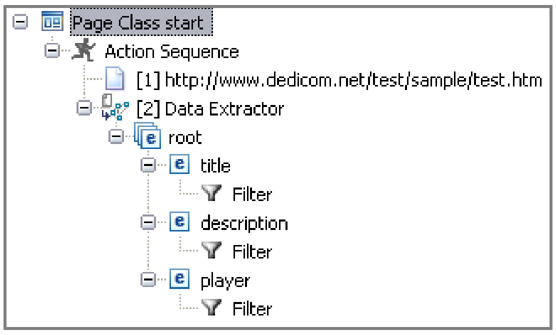
As we are going to extract three fields of the resulting search items we create the following data model with a root node at the top:



```
Data Model
  root
    title
    description
    player
```

Figure 24: Data model used by Lixto for the simple data extraction

This data model is used by Lixto to specify the format of the XML output. The next step consists of defining which the actions that Lixto should realize before extracting data are.



```
Page Class start
  Action Sequence
    [1] http://www.dedicom.net/test/sample/test.htm
    [2] Data Extractor
      root
        title
          Filter
        description
          Filter
        player
          Filter
```

Figure 25: Action sequence to extract data by Lixto

1. Go to the Web Page of our source data
2. Use a data extractor together with our data model and filters to extract the information

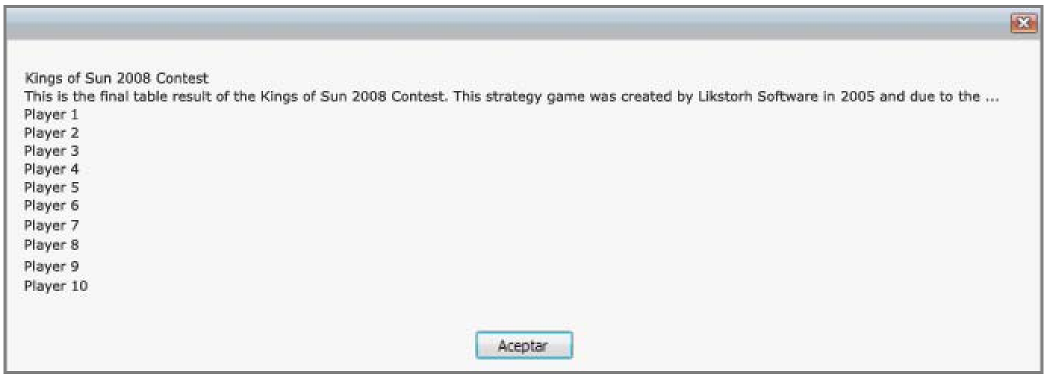
Once configured the filters to extract data we could extract all the fields correctly. Here is presented the result:

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
         <root>
            <title>Kings of Sun 2008 Contest</title>
            <description>This is the final table result of the Kings of Sun 2008
Contest. This strategy game was created by Likstorh Software in 2005 and due to the
growth of online players each year online competitions take place. The user has to use
his strategy habilities to be the best king of his land, that includes have a growing
population, construct temples, study new technologies, begin wars to extend
territory...</description>
            <player>Player 1</player>
            <player>Player 2</player>
            <player>Player 3</player>
            <player>Player 4</player>
            <player>Player 5</player>
            <player>Player 6</player>
            <player>Player 7</player>
            <player>Player 8</player>
            <player>Player 9</player>
            <player>Player 10</player>
         </root>
      </document>
```

■ WinTask:

To extract data with WinTask we have to edit a script file that will extract all the fields of interest. First of all, we need two orders, one to open the Internet explorer and other one to load the Web page source.

Then we only have to use the graphical interface to extract all the fields. No problems have been encountered with this tool and all the information has correctly been extracted.

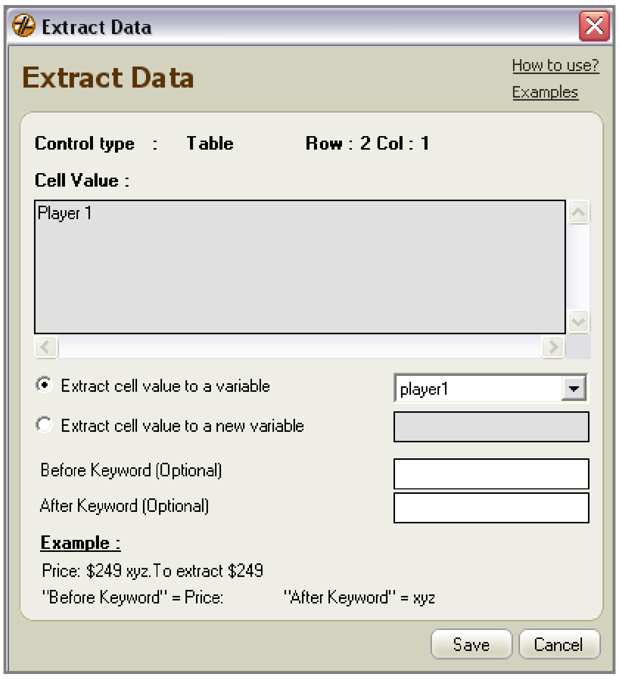


```
Kings of Sun 2008 Contest
This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the ...
Player 1
Player 2
Player 3
Player 4
Player 5
Player 6
Player 7
Player 8
Player 9
Player 10
```

Figure 26: Final output using WinTask

■ Automation Anywhere:

With Automation Anywhere we only have to create new variables to save the extracted values. Once created, we only have to select the specific content to extract and establish the relation to these variables. Then our results are extracted and can be outputted.



Extract Data

Control type : Table Row : 2 Col : 1

Cell Value :

Player 1

(●) Extract cell value to a variable: player1

( ) Extract cell value to a new variable

Before Keyword (Optional)

After Keyword (Optional)

Example :

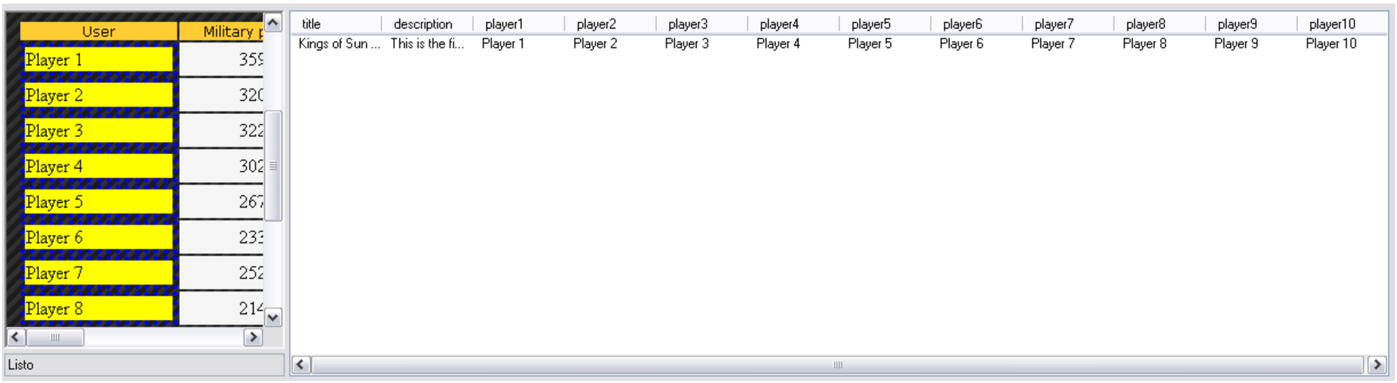
Price: $249 xyz.To extract $249

"Before Keyword" = Price: "After Keyword" = xyz

Figure 27: Saving a field with Automation Anywhere

■ Web Content Extractor:

Web Content Extractor presented no problems when extracting these fields. With this tool we only have to select the Web page source and select the fields we want to extract. We have to name each of the extracted fields to be referenced.



User	Military p
Player 1	359
Player 2	320
Player 3	322
Player 4	302
Player 5	267
Player 6	233
Player 7	252
Player 8	214

Listo

title	description	player1	player2	player3	player4	player5	player6	player7	player8	player9	player10
Kings of Sun ...	This is the fi...	Player 1	Player 2	Player 3	Player 4	Player 5	Player 6	Player 7	Player 8	Player 9	Player 10

Figure 28: Final output using Web Content Extractor

■ Goldseeker:

Using this tool to configure the data extraction process we have to edit the *sample.php* file:

```
<?
      include('GSparser.php');

      $dm = new GSParser('./kings.gs', './kings.data', 'singleFile');
      $dm->parse();
?>
```

In this file we indicate to include the *GSparser.php* which is the file that has all the tool functions and other source code. Then we use two files as parameters for the constructor of the GSParser:

- *Kings.data* which is the Web page containing all the HTML structure. In this case the file is directly provided by the local server.
- *Kings.gs* which contains the config files format. It configures the tool to extract data.

Without problems we extracted the data using Goldseeker, here we present a little part of the output.

```
Array
(
    [0] => Array
        (
            [name] => Title
            [instances] => Array
                (
                    [0] => Array
                        (
                            [contents] => Kings of Sun 2008 Contest
                            [position] => 1166
                        )
                )
          )
    [1] => Array
        (
            [name] => Description
            [instances] => Array
                (
                    [0] => Array
                        (
                            [contents] => This is the final table result of the Kings of
Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to
the growth of online players each year online competitions take place. The user has to
use his strategy habilities to be the best king of his land, that includes have a
growing population, construct temples, study new technologies, begin wars to extend
territory...
                            [position] => 1404
                        )
                )
        )
…
Done!
```

Figure 29: Final output using Goldseeker

■ Webharvest:

In this case we used Xpath expressions to extract data from our scenario.

The configuration file looks like that:

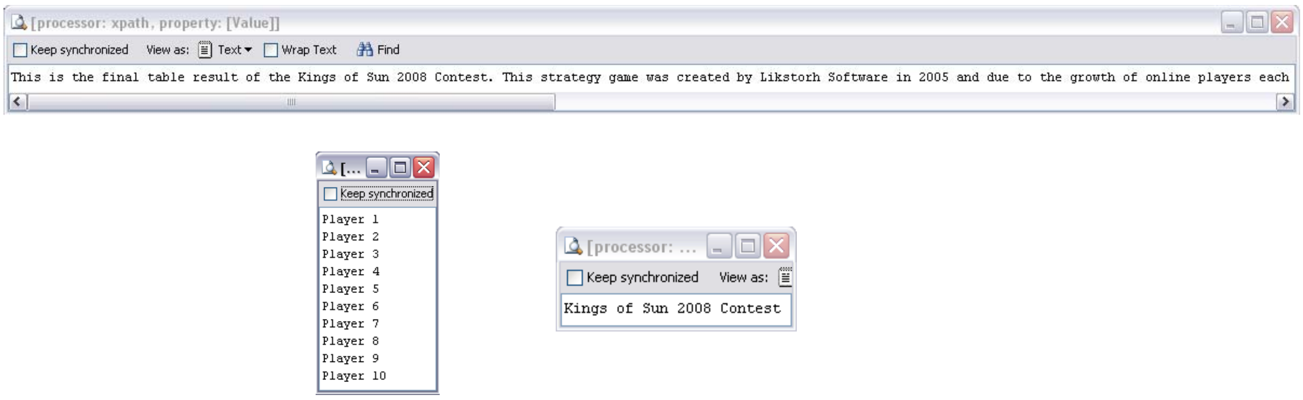
```
<?xml version="1.0" encoding="UTF-8"?>

<config charset="ISO-8859-1">


       <loop item="link" index="i" filter="unique">
              <list>
              <xpath expression="//div[matches(@class,'Estilo2$')]/text()">
                   <html-to-xml>
                       <http url="http://www.dedicom.net/test/sample/test.htm"/>
                   </html-to-xml>
               </xpath>
              <xpath expression="//div[contains(@class, 'Estilo6')]/div/text()">
                   <html-to-xml>
                       <http url="http://www.dedicom.net/test/sample/test.htm"/>
                   </html-to-xml>
               </xpath>
               <xpath expression="//td[contains(@height, '5')]/text()">
                   <html-to-xml>
                       <http url="http://www.dedicom.net/test/sample/test.htm"/>
                   </html-to-xml>
               </xpath>
           </list>
           <body>
                     <var name="link"></var>
           </body>
       </loop>

</config>
```

The first Xpath expression extracts the title from the Web page, the second one extracts the description and the third one all the names of the players. We have to use the information of the HTML tags to guide the tool to extract data. The results are shown here:



```
This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each
```

```
Player 1
Player 2
Player 3
Player 4
Player 5
Player 6
Player 7
Player 8
Player 9
Player 10
```

```
Kings of Sun 2008 Contest
```

Figure 30: Extracted data by Web-Harvest

In this example we have only used Xpath expressions to extract data but if we take a look to the manual section of the Web-Harvest Homepage we can find a big amount of functions that let us to perform more concrete actions like extracting data to files, transform HTML to XML or execute XQueries.

■ The following table summarizes the final results of our tests:

	Ebay search
Dapper	√
Robomaker	√
Lixto	√
WinTask	√
Automation Anywhere	√
Web Content Extractor	√
Goldseeker	√
Webharvest	√

Figure 31: Final basic data extractions

4.3.2- Data extraction from Web search engines

Web search engines are important for every day Web searches as they are a simple, fast and powerful way to find information of our interest. On those last years their popularity has been increasing and nowadays they are an indispensable tool.

We are going to use the three most used search engines that at the present we can find on the Web:

- Google
- Yahoo! Search
- Microsoft Live Search

Figure 32 illustrates the use percentage of them:

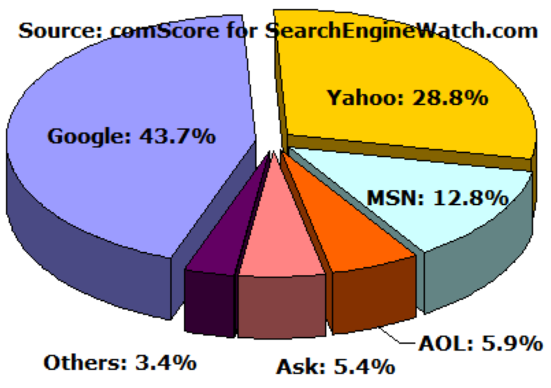


Figure 32: Percentage of use of the most important Web search engines – from [9]

This is a useful test as evaluate several features; to start we need an input value to perform a search and afterwards we receive a page with all the search results for our input value. This test is limited to the tools that let us to use an input value and to get data from a dynamic page content as the results change depending on the importance of the content, the number of searches and other factors. In another way we can use the GET value contained in the URL to perform a search in one step.

Maybe we can experience little data errors experimenting with some particular cases of input searches that produce a custom output, for example the description field of *YouTube* videos.

It is really important to select a good sample to realize the data extraction, we mean to include a varied structure of the output. Using Google if we use the input value of *Barcelona* we experience data loses in some cases as we don't include all the possible results. On the other hand, if we use the input value of *Lamborghini* the results have a more varied HTML structure that decrease the possibility of loosing data. But certainly in Web search engines we can not be 100% sure that all the data is going to be perfectly extracted as the result page doesn't have a static structure and it can experience changes.

In the next stage we are going to test each suitable tool with each search engine and comment the way that they perform the data extraction and the problems that we get.

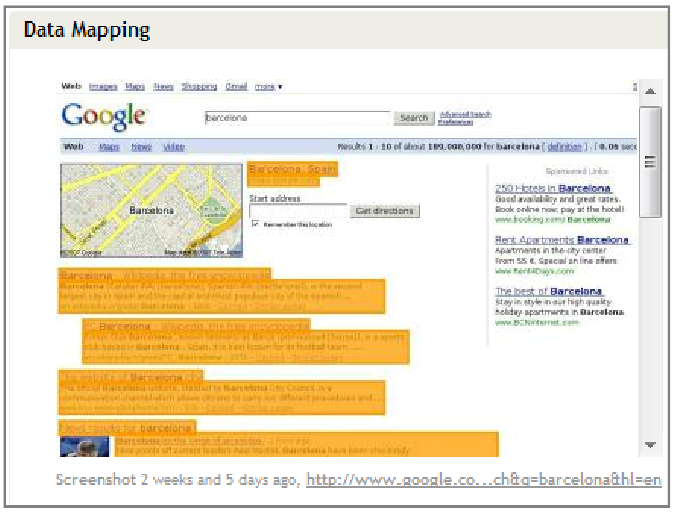
- **Dapper:**

Dapper let us to define an input variable to update the content of the search. This feature is really useful, if we want to realize another new search we only have to change the value of this input variable.

To configure the data extraction process we have to do a first search with an input variable and add the resulting page to the basket of sample pages.

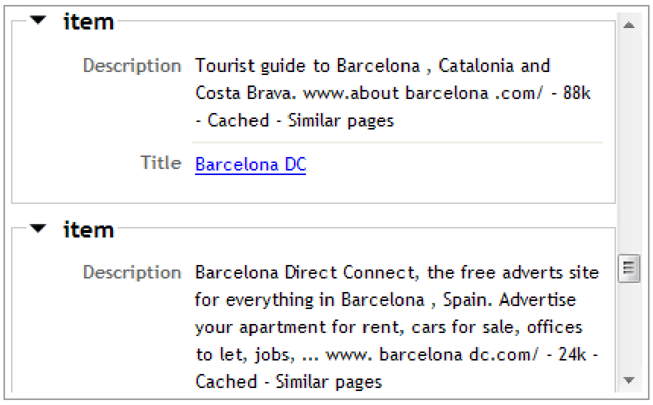
After that, we have to select interactively the fields of data that we want to extract, in this case all the information of each resulting entry. As explained before, we have to look for selecting a good sample for the input value. These tests have been built using the RSS feed output.

- **Google Search:** Dapper is able to extract all the entries without problems. It extracts the Google maps entries, normal links and nested links (see the third link of the left screenshot of figure 33). The description takes all the information: Text, cached content, size… It is suitable to be used to perform Google searches.



Data Mapping

Screenshot 2 weeks and 5 days ago, http://www.google.co....ch&q=barcelona&hl=en



item

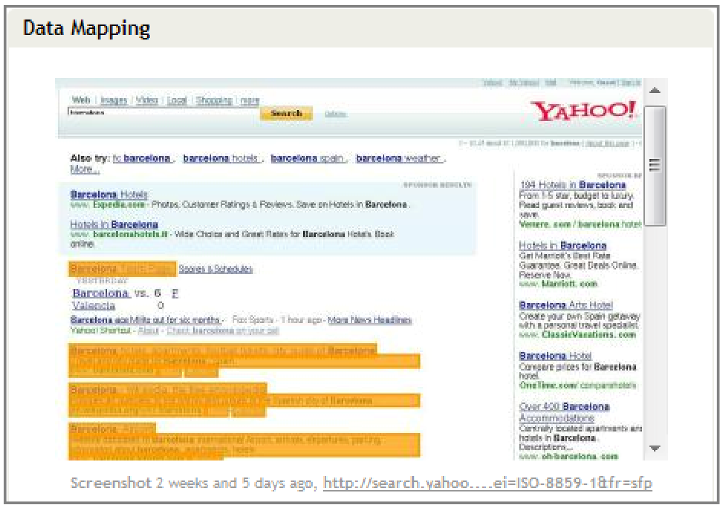
Description	Tourist guide to Barcelona , Catalonia and Costa Brava. www.about barcelona .com/ - 88k - Cached - Similar pages
Title	Barcelona DC

item

Description	Barcelona Direct Connect, the free adverts site for everything in Barcelona , Spain. Advertise your apartment for rent, cars for sale, offices to let, jobs, ... www. barcelona dc.com/ - 24k - Cached - Similar pages

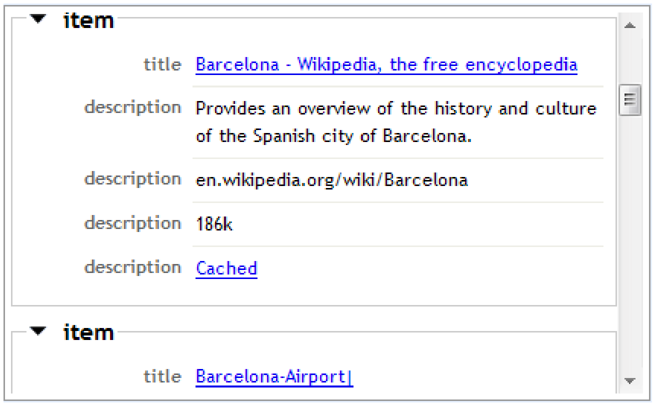
Figure 33: Google results with Dapper

- **Yahoo! Search:** In this case, Dapper is able to extract all the entries without problems. We have to mention that Yahoo! Search uses a live search input form with AJAX code, but in this case it doesn't affect to our data extraction. What happens is that we can not join all the description fields to a single description item because the HTML structure of the description does not include the previous commented extra fields (URL and cached) . In conclusion Dapper passes this test.



Data Mapping

Screenshot 2 weeks and 5 days ago, http://search.yahoo.....ei=ISO-8859-1&fr=sfp



item

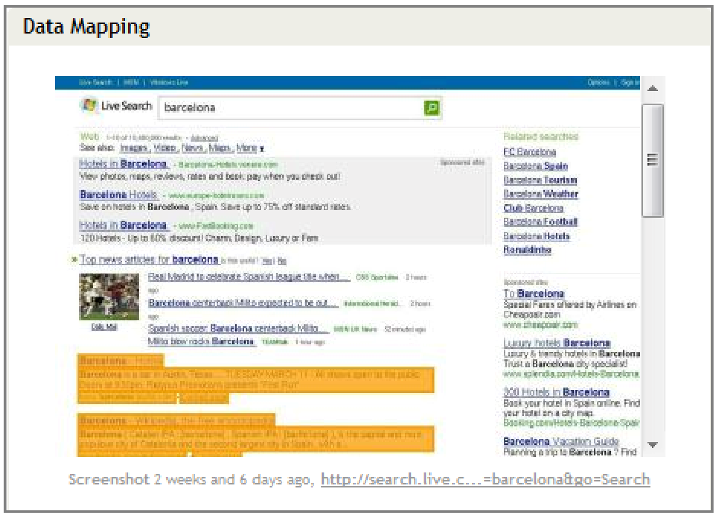
title	Barcelona - Wikipedia, the free encyclopedia
description	Provides an overview of the history and culture of the Spanish city of Barcelona.
description	en.wikipedia.org/wiki/Barcelona
description	186k
description	Cached

item

title	Barcelona-Airport|

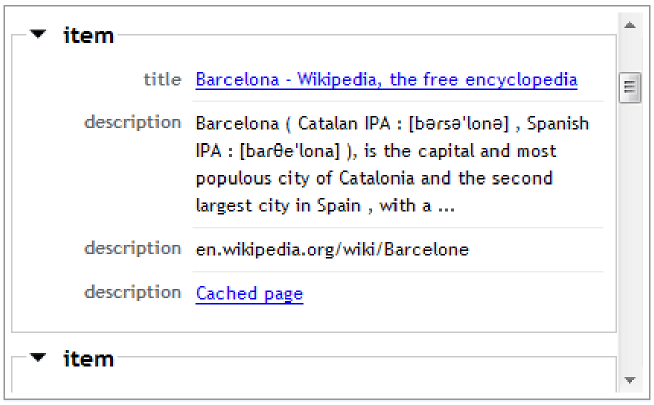
Figure 34: Yahoo! Search results with Dapper

- **Live Search:** We have extracted all the entries but as happened in Yahoo! Search we can not join all the description fields to a single description item. But we conclude that Dapper is able to perform searches through Microsoft Live Search without problems.



Data Mapping

Screenshot 2 weeks and 6 days ago, http://search.live.c...=barcelona&go=Search



item

title	Barcelona - Wikipedia, the free encyclopedia
description	Barcelona (Catalan IPA : [bərsə'lonə] , Spanish IPA : [barθe'lona]), is the capital and most populous city of Catalonia and the second largest city in Spain , with a ...
description	en.wikipedia.org/wiki/Barcelone
description	Cached page

item

Figure 35: MSN Live Search results with Dapper

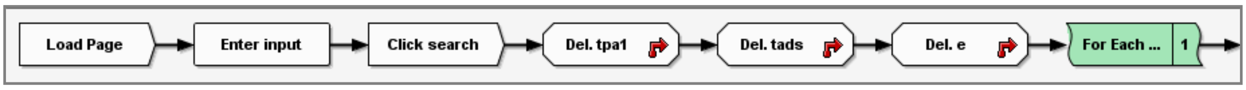
■ Robomaker:

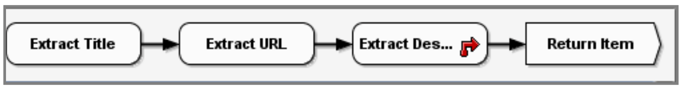
Like with Dapper we have chosen to use a RSS feed output. To realize a correct data extraction from these Web search engines we have to design a flow of actions that Robomaker sequentially executes.

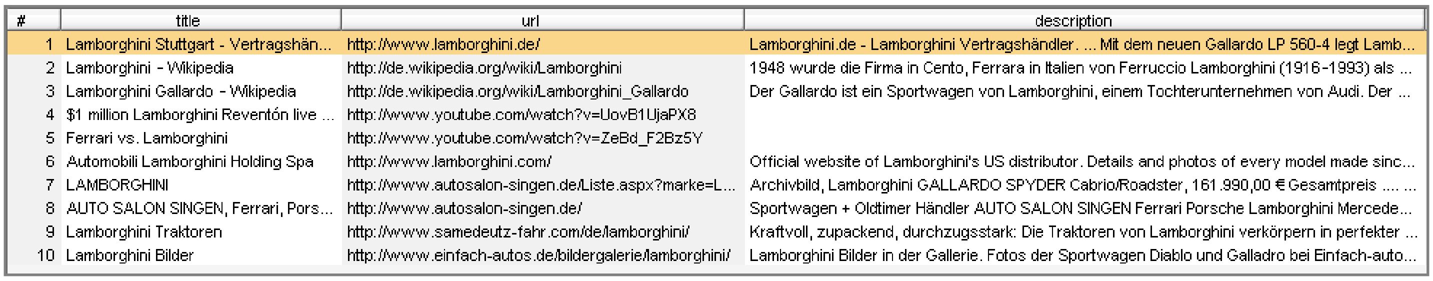
As we want to extract more than one element of our performed search we have to use a flow *step* that could iterate through the tag which identifies a resulting element. When we act in this way we sometimes experience problems. Together with the results, we encounter other annoying elements; such as sponsors, images or videos that don not have any interest for us. To avoid this we can use one of the Robomaker steps that allow us to remove these annoying tags before performing the data extraction. The highest possibility is that all this elements will not appear together, we have to ignore errors generated due to the absence of one of these tags.

Something similar happens when we iterate from entries that do not have the same structure. We have to perform more than one extraction step of an element afterwards to avoid data loses and for the same reason as before, we have to ignore produced errors.

- **Google:** Before extracting our data, we remove three types of annoying tags. The two first ones are from advertising and the third one refers to related images of the realized search. We iterate through the elements of the result container and extract the title, URL and description field of each resulting element. We have to ignore errors from the description field as we can find elements without description. This extraction includes all the information grouped (text, cached content, size…).





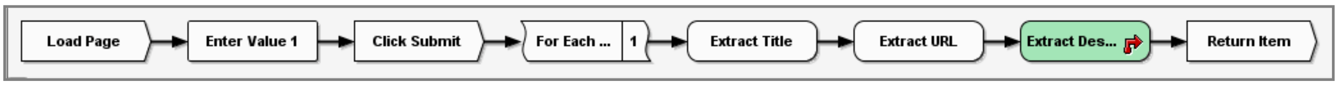


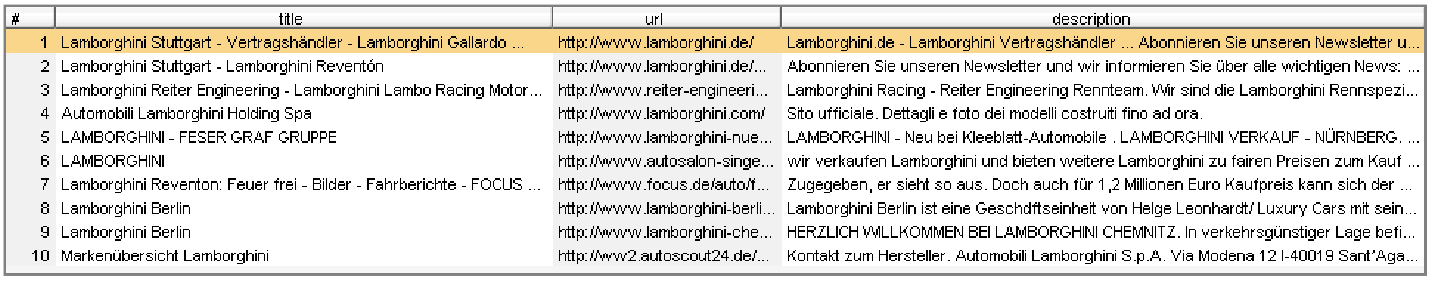
#	title	url	description
1	Lamborghini Stuttgart - Vertragshän...	http://www.lamborghini.de/	Lamborghini.de - Lamborghini Vertragshändler. ... Mit dem neuen Gallardo LP 560-4 legt Lamb...
2	Lamborghini - Wikipedia	http://de.wikipedia.org/wiki/Lamborghini	1948 wurde die Firma in Cento, Ferrara in Italien von Ferruccio Lamborghini (1916-1993) als ...
3	Lamborghini Gallardo - Wikipedia	http://de.wikipedia.org/wiki/Lamborghini_Gallardo	Der Gallardo ist ein Sportwagen von Lamborghini, einem Tochterunternehmen von Audi. Der ...
4	$1 million Lamborghini Reventón live ...	http://www.youtube.com/watch?v=UovB1UjaPX8	
5	Ferrari vs. Lamborghini	http://www.youtube.com/watch?v=ZeBd_F2Bz5Y	
6	Automobili Lamborghini Holding Spa	http://www.lamborghini.com/	Official website of Lamborghini's US distributor. Details and photos of every model made sinc...
7	LAMBORGHINI	http://www.autosalon-singen.de/Liste.aspx?marke=L...	Archivbild, Lamborghini GALLARDO SPYDER Cabrio/Roadster, 161.990,00 € Gesamtpreis ....
8	AUTO SALON SINGEN, Ferrari, Pors...	http://www.autosalon-singen.de/	Sportwagen + Oldtimer Händler AUTO SALON SINGEN Ferrari Porsche Lamborghini Mercede...
9	Lamborghini Traktoren	http://www.samedeutz-fahr.com/de/lamborghini/	Kraftvoll, zupackend, durchzugsstark: Die Traktoren von Lamborghini verkörpern in perfekter ...
10	Lamborghini Bilder	http://www.einfach-autos.de/bildergalerie/lamborghini/	Lamborghini Bilder in der Gallerie. Fotos der Sportwagen Diablo und Galladro bei Einfach-auto...

Figure 36: Google results with Robomaker

- **Yahoo! Search:** As we mentioned before this engine uses an AJAX live search in the input form of the searched item. Although we have a step in Robomaker to execute Javascript we could not receive the result page. For this reason, we could not extract any information from this page.

- **Live Search:** With the Microsoft Live Search we have directly a container that includes all the results. This means we do not have to worry about annoying elements. We can iterate directly through the result elements and get our desired data. As we can find results without description, we have to ignore the possible generated errors. As happened with Dapper, we do not take all the information elements from the description (URL, cached…)





#	title	url	description
1	Lamborghini Stuttgart - Vertragshändler - Lamborghini Gallardo ...	http://www.lamborghini.de/	Lamborghini.de - Lamborghini Vertragshändler ... Abonnieren Sie unseren Newsletter u...
2	Lamborghini Stuttgart - Lamborghini Reventón	http://www.lamborghini.de/...	Abonnieren Sie unseren Newsletter und wir informieren Sie über alle wichtigen News: ...
3	Lamborghini Reiter Engineering - Lamborghini Lambo Racing Motor...	http://www.reiter-engineeri...	Lamborghini Racing - Reiter Engineering Rennteam. Wir sind die Lamborghini Rennspezi...
4	Automobili Lamborghini Holding Spa	http://www.lamborghini.com/	Sito ufficiale. Dettagli e foto dei modelli costruiti fino ad ora.
5	LAMBORGHINI - FESER GRAF GRUPPE	http://www.lamborghini-nue...	LAMBORGHINI - Neu bei Kleeblatt-Automobile . LAMBORGHINI VERKAUF - NÜRNBERG. ...
6	LAMBORGHINI	http://www.autosalon-singe...	wir verkaufen Lamborghini und bieten weitere Lamborghini zu fairen Preisen zum Kauf ...
7	Lamborghini Reventon: Feuer frei - Bilder - Fahrberichte - FOCUS ...	http://www.focus.de/auto/f...	Zugegeben, er sieht so aus. Doch auch für 1,2 Millionen Euro Kaufpreis kann sich der ...
8	Lamborghini Berlin	http://www.lamborghini-berli...	Lamborghini Berlin ist eine Geschdftseinheit von Helge Leonhardt/ Luxury Cars mit sein...
9	Lamborghini Berlin	http://www.lamborghini-che...	HERZLICH WILLKOMMEN BEI LAMBORGHINI CHEMNITZ. In verkehrsgünstiger Lage befi...
10	Markenübersicht Lamborghini	http://ww2.autoscout24.de/...	Kontakt zum Hersteller. Automobili Lamborghini S.p.A. Via Modena 12 I-40019 Sant'Aga...

Figure 37: MSN Live Search results with Robomaker

■ Lixto:

As we are going to extract three fields of the resulting search items we create the following data model with a root node at the top:

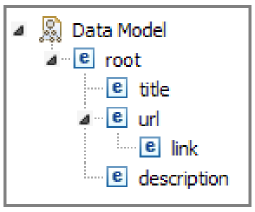


Figure 38: Data model used by Lixto for Web search engines tests

In this case we want to extract the title, the URL and the description of the result entries. From the URL we want to extract the link and not the text, this is why we use an internal node.

The next step consists of defining the actions that Lixto should realize before extracting data. As there are not many differences between our Web search engines we are going to explain this only for Google.

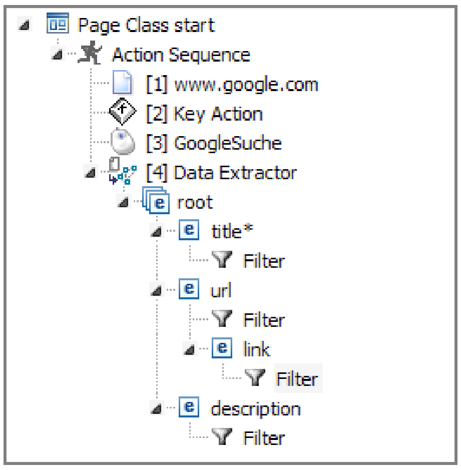


Figure 39: Action sequence to extract data by Lixto

1. Go to the Web Page of our selected search engine
2. Write the search value into the input form
3. Click to the *search* button
4. Use a data extractor together with our data model and filters to extract the information

- **Google:** All the data of the obtained results has been correctly extracted. We have to use a XPath expression to select both Google map entries and all the results together. Description includes all the information so we can say that Lixto works well with this Web search engine.

```
<lixto:extractor>
   <root>
      <title>Barcelona, Spanien</title>
      <url>
          <link>
               http://maps.google.de/maps?hl=de&amp;q=Barcelona,+Spanien&a
               mp;um=1&amp;ie=UTF-
               8&amp;sa=X&amp;oi=geocode_result&amp;resnum=1&amp;ct=title
          </link>
      </url>
      <description>maps.google.de</description>
   </root>
```

```
<root>
   <title>Barcelona – Wikipedia</title>
   <url>
      <link>http://de.wikipedia.org/wiki/Barcelona</link>
   </url>
   <description>
      Dieser Artikel behandelt die katalanische Stadt Barcelona; zu
      anderen gleichnamigen Bedeutungen siehe Barcelona
      (Begriffsklärung). ...de.wikipedia.org/wiki/Barcelona - 117k -
      Im Cache - Ähnliche Seiten
   </description>
</root>
 …
```

- **Yahoo! Search:** Using the same structure as with Google, Lixto is able to extract all the resulting information without taking data from the *sponsor* container. Description doesn't include all the information, but it works without problems.

```
<lixto:extractor>
    <root>
          <title>Barcelona hotels, apartments, football tickets, city guide of
          Barcelona
          </title>
          <url>
                 <link>http://rds.yahoo.com/_ylt=A0geu..cUyBIqxkBXRpXNyoA;_ylu=X3oD
                 MTEzZmF0YW5uBHNlYwNzcgRwb3MDMQRjb2xvA2FjMgR2dGlkA0RGRDVfMTA1/SIG=1
                 1djgduo4/EXP=1210164508/**http%3a//www.barcelona.com/</link>
         </url>
       <description>Travel and city guide for Barcelona, Spain.</description>
    </root>
    <root>
          <title>Barcelona - Wikipedia, the free encyclopedia</title>
          <url>
                 <link>http://rds.yahoo.com/_ylt=A0geu..cUyBIqxkBXxpXNyoA;_ylu=X3oD
                 MTEzbTRiNWs0BHNlYwNzcgRwb3MDMgRjb2xvA2FjMgR2dGlkA0RGRDVfMTA1/SIG=1
                 1qlukckh/EXP=1210164508/**http%3a//en.wikipedia.org/wiki/Barcelona
                 </link>
          </url>
          <description>Provides an overview of the history and culture of the
          Spanish city of Barcelona.</description>
    </root>
    ...
```

- **Live Search:** Lixto is able to extract all the requested data avoiding videos and sponsors content. As happens with Yahoo! Search, description doesn't take all the complete information. So we can conclude that Lixto works without problems with Microsoft Live Search.

```
<lixto:extractor>
    <root>
          <title>Barcelona.de - Reiseführer. Hotel, Flug, Barcelona Card
          buchen</title>
            <url>
               <link>http://www.barcelona.de/</link>
            </url>
                 <description>Information über die Hauptstadt Kataloniens. Mit
                 Hinweisen zu Sehenswürdigkeiten, Hotels, Gastronomie, Kunst und
                 Kultur, Natur und Umgebung. Zusätzlich gibt es ein
                 ...</description>
    </root>
    <root>
       <title>*Barcelona.de - Hotel, Flug und Mietwagen buchen</title>
       <url>
          <link>http://www.barcelona.de/de/2.php</link>
       </url>
          <description>Barcelona, kulturelle Hauptstadt Spaniens. Ein Reisef|hrer.
          ... Sie erhalten auf dieser Seite einen Überblick über die vielfältigen
          Sehenswürdigkeiten in Barcelona.</description>
    </root>

    …
```

■ WinTask:

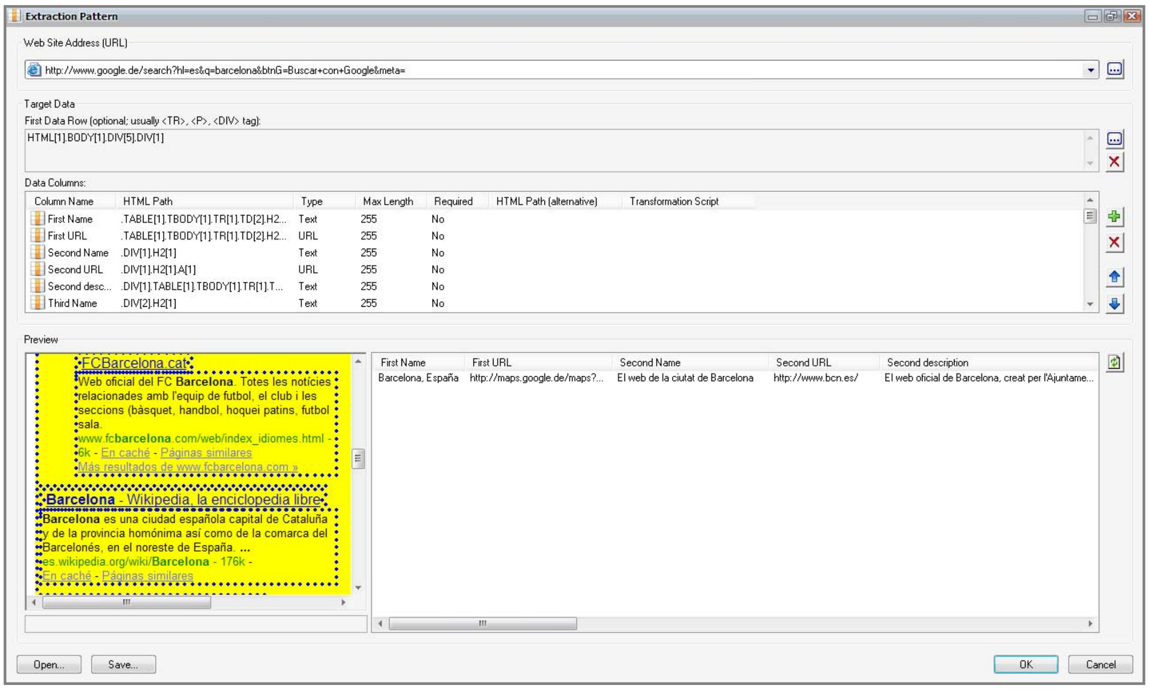
WinTask can use HTML descriptors to detect data that we want to search in the document and then realize an extraction. It is not able to extract dynamic information as its engine works using the name of the container that has the information that we want to extract. This represents a problem as it is focused to treat with static content. With this tool the user is able to write complex scripts. However it is basically built to work with automation of tasks, so we can not take advantage of it in Web search engines.

■ Automation Anywhere:

This tool let the user record a set of actions to perform a customized search. To realize extractions in the Web search engines field we don't have enough resources by the tool and we experience difficulties. This tool, like Wintask, it is more oriented on the task automation field. For this reason we could not extract all the data as well.

■ Web Content Extractor:

This tool doesn't allow the user to insert information in a HTML form and pick up the resulting data after clicking the submit button. We have to use a GET method to generate directly a result page. When talking about the data extraction it is able to choose the tag that contains information and save it in a result column. It doesn't use iterators, it is only guided by the tags that the page contains. It is very slow to manually develop this work but it guarantees the correctness of the extracted data as it uses the page structure. As we have to select element by element the data this program will work in all the Web search engines, the problem it is that configuring the extraction could require so much time. In conclusion, this tool passes the test.



Extraction Pattern

Web Site Address (URL)

http://www.google.de/search?hl=es&q=barcelona&btnG=Buscar+con+Google&meta=

Target Data

First Data Row (optional; usually <TR>, <P>, <DIV> tag):

HTML[1].BODY[1].DIV[5].DIV[1]

Data Columns:

Column Name	HTML Path	Type	Max Length	Required	HTML Path (alternative)	Transformation Script
First Name	.TABLE[1].TBODY[1].TR[1].TD[2].H2...	Text	255	No		
First URL	.TABLE[1].TBODY[1].TR[1].TD[2].H2...	URL	255	No		
Second Name	.DIV[1].H2[1]	Text	255	No		
Second URL	.DIV[1].H2[1].A[1]	URL	255	No		
Second desc...	.DIV[1].TABLE[1].TBODY[1].TR[1].T...	Text	255	No		
Third Name	.DIV[2].H2[1]	Text	255	No		

Preview

FCBarcelona.cat
Web oficial del FC **Barcelona**. Totes les notícies relacionades amb l'equip de futbol, el club i les seccions (bàsquet, handbol, hoquei patins, futbol sala.
www.fc**barcelona**.com/web/index_idiomes.html - 6k - En caché - Páginas similares
Más resultados de www.fcbarcelona.com »

Barcelona - Wikipedia, la enciclopedia libre
Barcelona es una ciudad española capital de Cataluña y de la provincia homónima así como de la comarca del Barcelonés, en el noreste de España. **...**
es.wikipedia.org/wiki/**Barcelona** - 176k -
En caché - Páginas similares

First Name	First URL	Second Name	Second URL	Second description
Barcelona, España	http://maps.google.de/maps?...	El web de la ciutat de Barcelona	http://www.bcn.es/	El web oficial de Barcelona, creat per l'Ajuntame...

Figure 40: Web Content Extractor results screenshot

■ Goldseeker:

With Goldseeker the data extraction process is based using a substring method which is configured into a file. To realize an extraction we need the name of the container/tag that contains the information that we want to extract and depending of the realized search this result is going to vary, so it is focused to treat with static content. We can not take advantage of it in Web search engines.

■ Webharvest:

To realize an extraction with Webharvest we have to edit an entire configuration file. We have used Xpath expressions to extract all the different fields from Web Search Engines results.

The process to extract information consists of looking into the HTML code and search for the specific tags and attributes that identifies each of the fields.

We are going to analyze together the three different types of Web search engines as the way to proceed is similar and only some tags from the configuration file vary.

The following code lines extract all the data fields from the Google search:

```
<?xml version="1.0" encoding="UTF-8"?>

<config charset="UTF-8">
    <var-def name="search" overwrite="false">barcelona</var-def>

      <var-def name="all">
        <xpath   expression="//h3[contains(@class,'r')]/*/text()   |   //a/em/text()   |
       //h3/a[contains(@class,'l')]/@href    |    //div[contains(@class,'s')]/text()    |
       //em/text()">
            <html-to-xml>
               <http
               url="http://www.google.com/search?hl=es&amp;q=${search}&amp;btnG=Buscar+c
               on+Google&amp;lr=lang_en"/>
            </html-to-xml>
        </xpath>
    </var-def>
</config>
```

Here is displayed a share of the output given by Webharvest when extracting data from the Google search:

```
http://maps.google.com/maps?hl=es&q=barcelona&lr=lang_en&um=1&ie=UTF-
8&sa=X&oi=geocode_result&resnum=1&ct=title

http://www.fcbarcelona.com/
FCBarcelona.cat
fcbarcelona.cat.

http://www.fcbarcelona.com/web/english/
FCBarcelona.cat
The official FCBarcelona website. All the latest news about the club’s football team and
the various sporting sections (Basketball Handball, Roller Hockey,

http://www.zoobarcelona.com/
Parc Zoològic de Barcelona, S.A.
Fichas de animales, revista, visita virtual y webcams de algunos animales en directo.
```

The only problem that occurred is that the data has been retrieved following another order as expected. Using the other two Web search engines the results were successful and no problems have occurred.

■ The following table summarizes the final results of our tests:

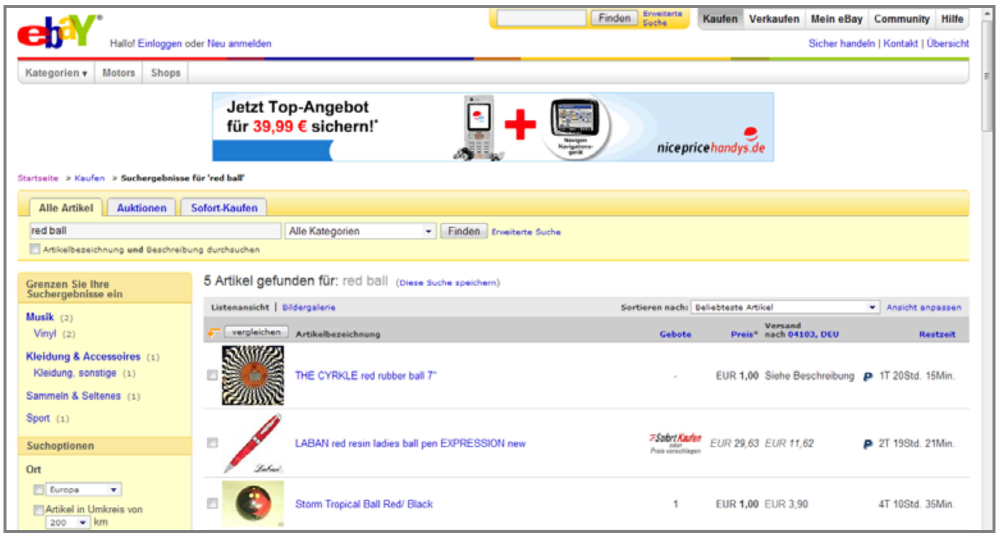
	Google Search	Yahoo! Search	MS Live Search
Dapper	√	√	√
Robomaker	√	X	√
Lixto	√	√	√
WinTask	X	X	X
Automation Anywhere	X	X	X
Web Content Extractor	√	√	√
Goldseeker	X	X	X
Webharvest	√	√	√

Figure 41: Final Web search engines test results

4.3.2- Data extraction from Ebay

The second test that we are going to execute with our data extraction tools consists of extracting data from an Ebay product search. It is the most important auction shop of the Web and it is famous all over the world. This test could be useful because of the use of input values and non static result pages. The information is organized in fields (rows and columns of data, which is the most outstanding feature). For each resulting product we are going to extract the next fields:

- Product name
- Price
- Shipping price
- Remaining time



Finden
Erweiterte Suche
Kaufen
Verkaufen
Mein eBay
Community
Hilfe
Hallo! Einloggen oder Neu anmelden
Sicher handeln | Kontakt | Übersicht
Kategorien
Motors
Shops
Jetzt Top-Angebot für 39,99 € sichern!*
nicepricehandys.de
Startseite > Kaufen > Suchergebnisse für 'red ball'
Alle Artikel
Auktionen
Sofort-Kaufen
red ball
Alle Kategorien
Finden
Erweiterte Suche
Artikelbezeichnung und Beschreibung durchsuchen
Grenzen Sie Ihre Suchergebnisse ein
Musik (2)
Vinyl (2)
Kleidung & Accessoires (1)
Kleidung, sonstige (1)
Sammeln & Seltenes (1)
Sport (1)
Suchoptionen
Ort
Europa
Artikel in Umkreis von
200 km
Listenansicht | Bildergalerie
Sortieren nach: Beliebteste Artikel
Ansicht anpassen
vergleichen

5 Artikel gefunden für: red ball (Diese Suche speichern)

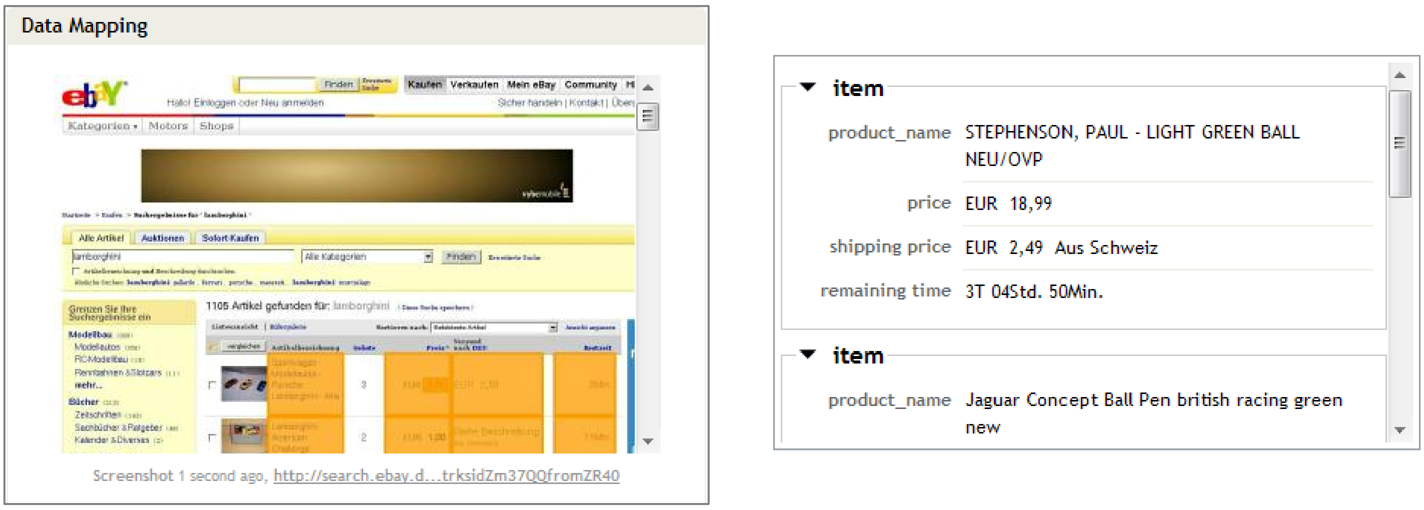
Artikelbezeichnung	Gebote	Preis*	Versand nach 04103, DEU		Restzeit
THE CYRKLE red rubber ball 7"	-	EUR 1,00	Siehe Beschreibung	P	1T 20Std. 15Min.
LABAN red resin ladies ball pen EXPRESSION new	Sofort Kaufen oder Preis vorschlagen	EUR 29,63	EUR 11,62	P	2T 19Std. 21Min.
Storm Tropical Ball Red/ Black	1	EUR 1,00	EUR 3,90		4T 10Std. 35Min.

Figure 42: Resulting page of an Ebay search

To execute this test we have to consider that if a search is not so specific then we retrieve a more general page with categories that don't follow the same structure. This could be a problem because we don't always receive results.

■ Dapper:

After realizing some executions with different values, sometimes happened that for some unknown reasons searches that return some correct results have not been extracted by Dapper. We conclude that Dapper occasionally experience problems to extract information from the Ebay product search.



Data Mapping
Kaufen
Verkaufen
Mein eBay
Community
Kategorien
Motors
Shops
Alle Artikel
Auktionen
Sofort Kaufen
Finden
1105 Artikel gefunden für: lamborghini
Screenshot 1 second ago, http://search.ebay.d...trksidZm37QQfromZR40

item

product_name	STEPHENSON, PAUL - LIGHT GREEN BALL NEU/OVP
price	EUR 18,99
shipping price	EUR 2,49 Aus Schweiz
remaining time	3T 04Std. 50Min.

item

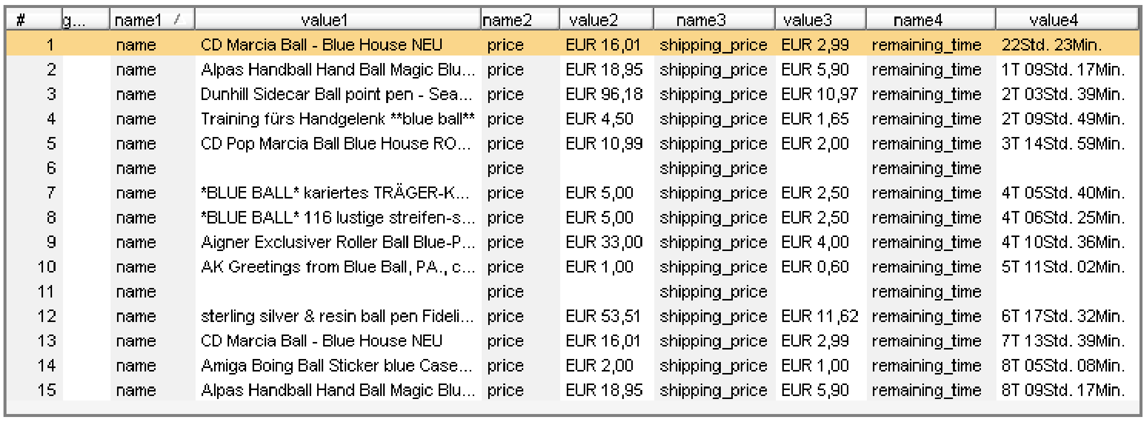
product_name	Jaguar Concept Ball Pen british racing green new

Figure 43: Ebay results with Dapper

■ Robomaker:

With Robomaker we have actions to solve the specialization grade problem of the search, for example using branches.

It detects two different types of prices at the data extraction time, so we use two different steps to realize the extraction although we are going to have a single result depending of the structure of the price field. We have to ignore errors as empty rows that don't contain information can appear.



#	g...	name1 /	value1	name2	value2	name3	value3	name4	value4
1		name	CD Marcia Ball - Blue House NEU	price	EUR 16,01	shipping_price	EUR 2,99	remaining_time	22Std. 23Min.
2		name	Alpas Handball Hand Ball Magic Blu...	price	EUR 18,95	shipping_price	EUR 5,90	remaining_time	1T 09Std. 17Min.
3		name	Dunhill Sidecar Ball point pen - Sea...	price	EUR 96,18	shipping_price	EUR 10,97	remaining_time	2T 03Std. 39Min.
4		name	Training fürs Handgelenk **blue ball**	price	EUR 4,50	shipping_price	EUR 1,65	remaining_time	2T 09Std. 49Min.
5		name	CD Pop Marcia Ball Blue House RO...	price	EUR 10,99	shipping_price	EUR 2,00	remaining_time	3T 14Std. 59Min.
6		name		price		shipping_price		remaining_time	
7		name	*BLUE BALL* kariertes TRÄGER-K...	price	EUR 5,00	shipping_price	EUR 2,50	remaining_time	4T 05Std. 40Min.
8		name	*BLUE BALL* 116 lustige streifen-s...	price	EUR 5,00	shipping_price	EUR 2,50	remaining_time	4T 06Std. 25Min.
9		name	Aigner Exclusiver Roller Ball Blue-P...	price	EUR 33,00	shipping_price	EUR 4,00	remaining_time	4T 10Std. 36Min.
10		name	AK Greetings from Blue Ball, PA., c...	price	EUR 1,00	shipping_price	EUR 0,60	remaining_time	5T 11Std. 02Min.
11		name		price		shipping_price		remaining_time	
12		name	sterling silver & resin ball pen Fideli...	price	EUR 53,51	shipping_price	EUR 11,62	remaining_time	6T 17Std. 32Min.
13		name	CD Marcia Ball - Blue House NEU	price	EUR 16,01	shipping_price	EUR 2,99	remaining_time	7T 13Std. 39Min.
14		name	Amiga Boing Ball Sticker blue Case...	price	EUR 2,00	shipping_price	EUR 1,00	remaining_time	8T 05Std. 08Min.
15		name	Alpas Handball Hand Ball Magic Blu...	price	EUR 18,95	shipping_price	EUR 5,90	remaining_time	8T 09Std. 17Min.

Figure 44: Ebay results with Robomaker

■ Lixto:

As we are going to extract four fields of the resulting search items we create the following data model with a root node at the top:

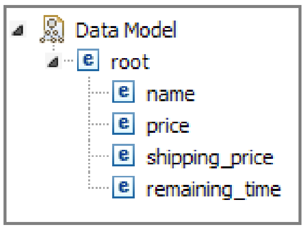


Figure 45: Data model used by Lixto for Ebay tests

With this structure the actions that we are going to perform to extract data with Lixto are the following.

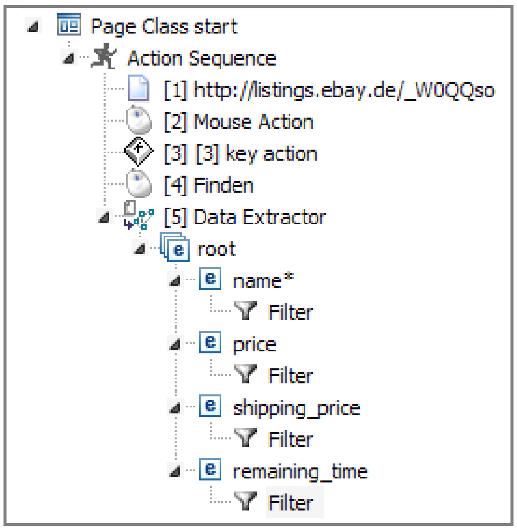


Figure 46: Action sequence to extract data by Lixto

1. Go to the Web Page of Ebay
2. Click to the input form
3. Write the product value into the input form
4. Click to the *search* button
5. Use a data extractor together with our data model and filters to extract the information

We get the result page in the same way as dapper or Robomaker, we get a category page or a product page depending on the input product. If the search doesn't output any result, then the tags of our XML file are empty. Nevertheless the data has been extracted without problems.

```
<lixto:extractor>
         <root>
            <name>CD Marcia Ball - Blue House NEU </name>
            <price>EUR 16,01</price>
```

```
      <shipping_price>EUR 2,99 </shipping_price>
      <remaining_time>21Std. 57Min.</remaining_time>
   </root>
   <root>
      <name>Alpas Handball Hand Ball Magic Blue 777 Handbälle Bälle </name>
      <price>EUR 18,95</price>
      <shipping_price>EUR 5,90 </shipping_price>
      <remaining_time>1T 08Std. 51Min.</remaining_time>
   </root>
   ...
```

■ WinTask:

As the Ebay search structure is similar to the one of Web search engines we have the same problem as in the previous test. This tool doesn't allow the user to handle dynamic content correctly, it only works fine with static content.

■ Automation Anywhere:

With this tool we arrive to the same conclusion as using Web search engines.

■ Web Content Extractor:

With this tool we arrive to the same conclusion as using Web search engines.

■ Goldseeker:

With this tool we arrive to the same conclusion as using Web search engines.

■ Webharvest:

We have extracted successfully all the fields presented in the Ebay search using Xpath expressions again.

The process of extracting the information consists of looking into the HTML code and search for the specific tags and attributes that identifies each of the fields that we want to extract.

These following code lines show the configuration file used to realize the extraction:

```
<?xml version="1.0" encoding="UTF-8"?>

<config charset="UTF-8">

    <var-def name="search" overwrite="false">redQ20ball</var-def>

    <var-def name="all">
      <xpath             expression="//div[contains(@class,'ttl')]/*/text()            |
      //div[contains(@class,'g-b')]/text()  |  //span[contains(@class,'ship  fee')]/text()
      | //span[contains(@class,'time')]/text() ">
       <html-to-xml>
         <http
         url="http://shop.ebay.de/items/_W0QQ_nkwZ${search}QQ_armrsZ1QQ_fromZQQ_mdoZ"/>
       </html-to-xml>
       </xpath>
    </var-def>

</config>
```

As happened with the Web search engines tests the limitation of the extraction depends of the limitation of the formed Xpath expressions. In this case Ebay presented no problems to perform the extraction. A part of the final output is shown next:

```
Cyrkle, The - Red Rubber Ball (A Collection) CD OVP new
EUR 8,49
+EUR 2,90
13Std 13Min

RED HOT N´ BLUE - Havin´ a Ball in RED VINYL RARE !!!!!
EUR 1,00
+EUR 4,00
2T 12Std 34Min

Neil Diamond - La Bamba / Red Rubber Ball - 1973  7"
EUR 2,49
+EUR 1,80
2T 17Std 9Min

THE CYRKLE - RED RUBBER BALL, #1 HIT USA APRIL 1966
EUR 1,00
+EUR 2,20
4T 22Std 32Min
```

- The following table summarizes the final results of our tests:

	Ebay search
Dapper	√ / X
Robomaker	√
Lixto	√
WinTask	X
Automation Anywhere	X
Web Content Extractor	√
Goldseeker	X
Webharvest	√

Figure 47: Final Ebay test results

4.3.3- Data extraction from dynamic content Web pages

Pageflakes

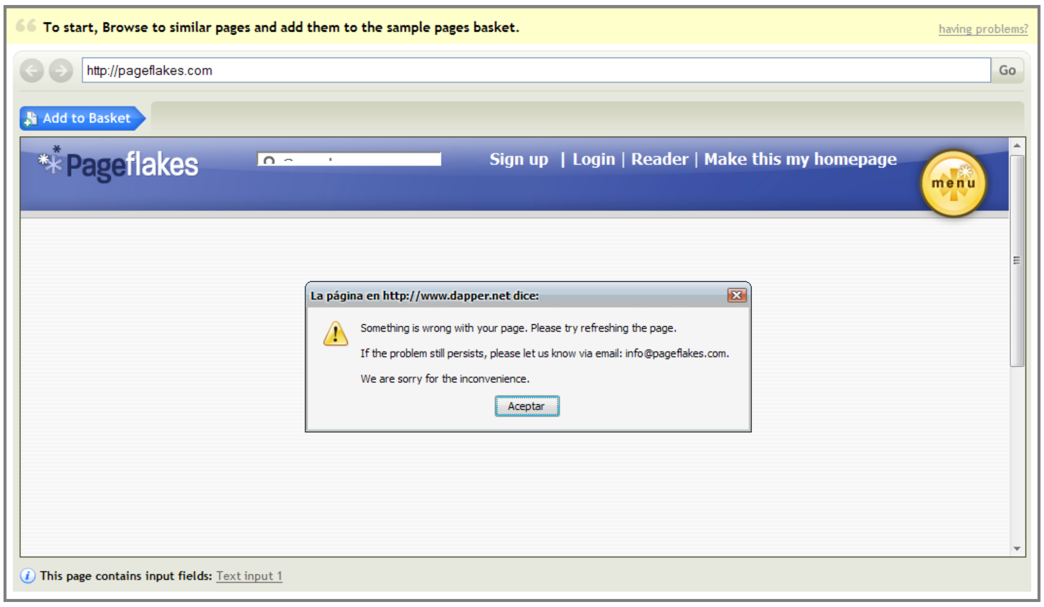
The aim of Pageflakes is to create a personalized Web page where all his users can keep up to date many blogs and new sources that are going to be read frequently.

This stage was chosen to test our tools with dynamic content Web pages using ASP.NET, AJAX or Javascript content. As Pageflakes is mainly constructed using this kind of content, it has been selected to be an applicant Web page to extract data.

We are going to extract data from the weather widget, specifically four fields; the first and the second name of the day and its weather information.

■ Dapper:

We got an error trying Dapper to extract data at the time of collecting sample pages. This error means our tool do not to pass this test.


To start, Browse to similar pages and add them to the sample pages basket.

La página en http://www.dapper.net dice:

Something is wrong with your page. Please try refreshing the page.

If the problem still persists, please let us know via email: info@pageflakes.com.

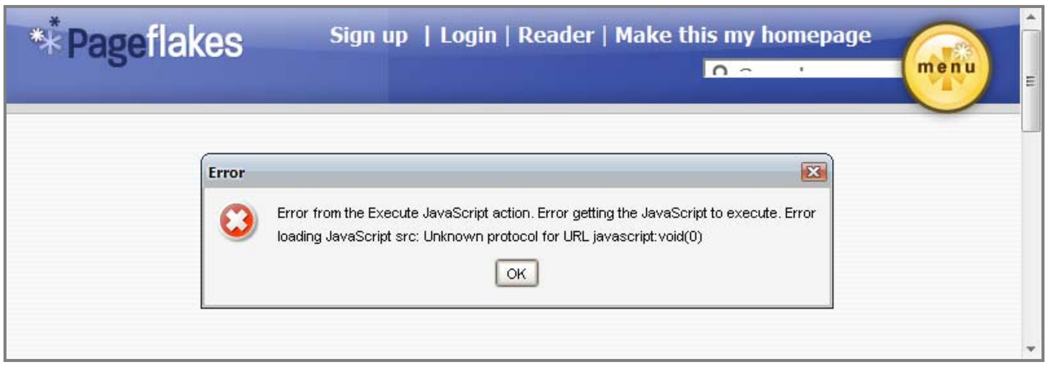
We are sorry for the inconvenience.

This page contains input fields: Text input 1

Figure 48: Pageflakes results with Dapper

■ Robomaker:

With Robomaker we can not load the start page of Pageflakes. To solve this problem we have tried to execute all the Javascript content of the page using the *Execute Javascript* step but after doing that we received another error. So this tool fails to extract data from this Web page too.


Error

Error from the Execute JavaScript action. Error getting the JavaScript to execute. Error loading JavaScript src: Unknown protocol for URL javascript:void(0)

Figure 49: Pageflakes results with Robomaker

■ Lixto:

With Lixto all the dynamic resources don't load correctly and we can not extract our desired data. It fails this test.

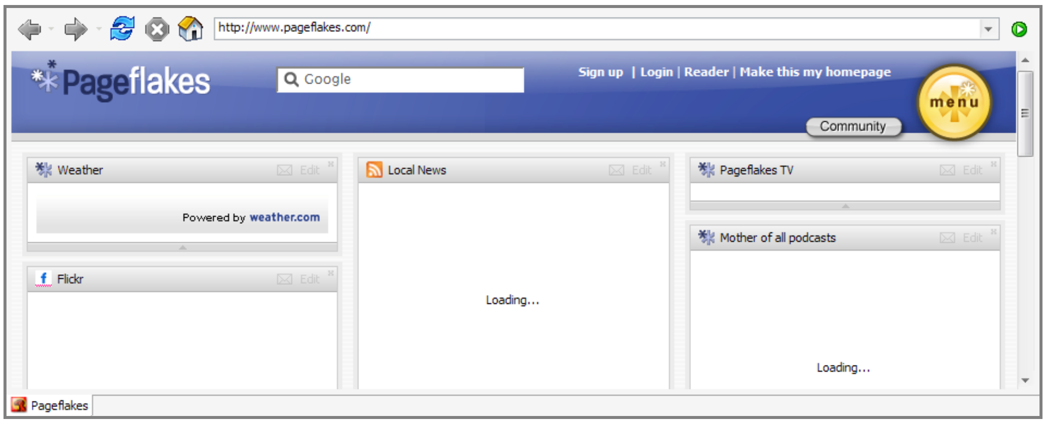


Figure 50: Pageflakes results with Lixto

■ WinTask:

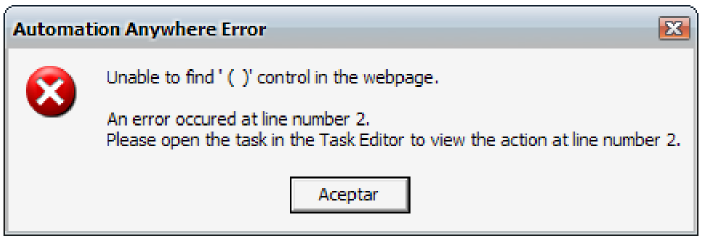
WinTask as it uses directly the Internet browser. For this reason it is able to display the dynamic page content without problems. The problem appears when we want to extract data, an error message of the data we want to extract cannot be found. This happens as it needs the content of the tag to extract the data correctly. For example from the temperature:

```
CaptureHTML("DIV[CONTENT='69']",captured_string$)
```

In this case, when we experience a change to the temperature value, the content of the tag changes too and then this produces an error. In conclusion, WinTask doesn't pass this test.

■ Automation Anywhere:

As it happens with WinTask, Automation Anywhere is able to record the actions that we perform in our browser and is able to show the dynamic content too. We have used 4 variables to extract dynamic information without problems. It worked fine for one day, but when the next day the information was renewed we experienced problems.


Automation Anywhere Error

Unable to find ' ( )' control in the webpage.

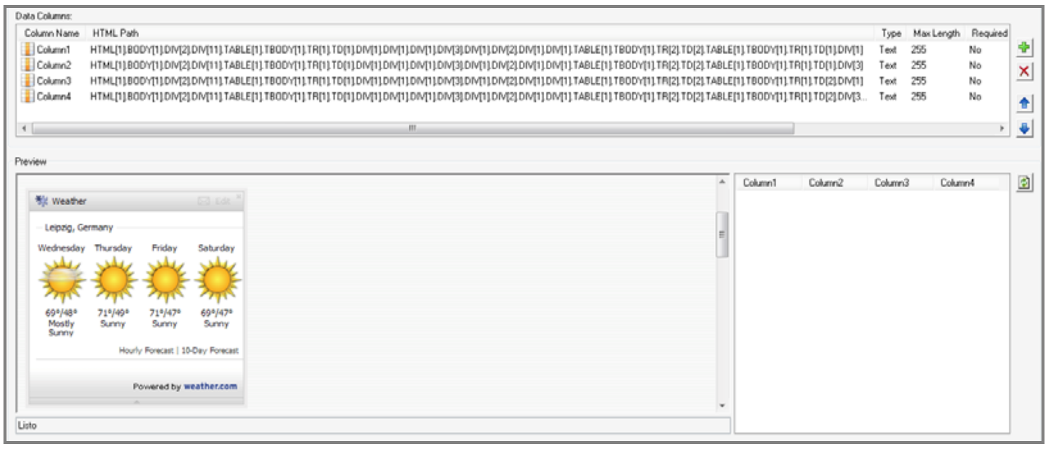
An error occured at line number 2.
Please open the task in the Task Editor to view the action at line number 2.

Aceptar

Figure 51: Pageflakes results with Automation Anywhere

■ Web Content Extractor:

This program allows us to extract information of dynamic Web pages. We have extracted 3 of the 4 fields that we wanted, the last one didn't return in any result. Executing the same task the day after we didn't retrieve any result. Then we can conclude that this program doesn't work correctly with this type of dynamic content.


Data Columns:

Column Name	HTML Path	Type	Max Length	Required
Column1	HTML[1].BODY[1].DIV[2].DIV[11].TABLE[1].TBODY[1].TR[1].TD[1].DIV[1].DIV[1].DIV[1].DIV[3].DIV[1].DIV[2].DIV[1].DIV[1].TABLE[1].TBODY[1].TR[2].TD[2].TABLE[1].TBODY[1].TR[1].TD[1].DIV[1]	Text	255	No
Column2	HTML[1].BODY[1].DIV[2].DIV[11].TABLE[1].TBODY[1].TR[1].TD[1].DIV[1].DIV[1].DIV[1].DIV[3].DIV[1].DIV[2].DIV[1].DIV[1].TABLE[1].TBODY[1].TR[2].TD[2].TABLE[1].TBODY[1].TR[1].TD[1].DIV[3]	Text	255	No
Column3	HTML[1].BODY[1].DIV[2].DIV[11].TABLE[1].TBODY[1].TR[1].TD[1].DIV[1].DIV[1].DIV[1].DIV[3].DIV[1].DIV[2].DIV[1].DIV[1].TABLE[1].TBODY[1].TR[2].TD[2].TABLE[1].TBODY[1].TR[1].TD[2].DIV[1]	Text	255	No
Column4	HTML[1].BODY[1].DIV[2].DIV[11].TABLE[1].TBODY[1].TR[1].TD[1].DIV[1].DIV[1].DIV[1].DIV[3].DIV[1].DIV[2].DIV[1].DIV[1].TABLE[1].TBODY[1].TR[2].TD[2].TABLE[1].TBODY[1].TR[1].TD[2].DIV[3...	Text	255	No

Preview

Column1	Column2	Column3	Column4

Figure 52: Pageflakes results with Web Content Extractor

■ Goldseeker:

With this tool we arrive to the same conclusion as using Web search engines and Ebay. This tool doesn't pass this test.

■ Webharvest:

With this tool, as we don't have to display the dynamic information we don't face any problem referred to the presentation of the content. We can directly construct a Xpath expression to extract the desired data.

What happened in this case is that we could not find a Xpath expression to extract the desired fields because we didn't have enough information from the tags to refer the fields of our interest.

The interesting data is placed between a div tag with no attributes. Due to this fact, we can not extract the data correctly from Pageflakes as we receive more data as desired.

```
<div>Tuesday</div>

<div><img src="Pageflakes_files/33.png" width="63" border="0" height="63"></div>

<div>13°C<br></div></td><td  style="width:  25%;  vertical-align:  top;  text-align:
center;"><div>Wednesday</div>

<div><img src="Pageflakes_files/11.png" width="63" border="0" height="63"></div>

<div>23°/13°C<br><span style="line-height: 100%;"></span></div>
```

■ The following table summarizes the final results of our tests:

	Pageflakes
Dapper	X
Robomaker	X
Lixto	X
WinTask	X
Automation Anywhere	X
Web Content Extractor	X
Goldseeker	X
Webharvest	X

Figure 53: Final Pageflakes test results

4.4- Resilience against changing HTML code

One of the main characteristics of the data extracting tools is that they use the structure of the HTML code to locate information and afterwards extract it.

It is very common that a Web page structure varies to extend the content, to improve the visual design or to introduce new Web technologies. All these changes could produce loses of data or errors to our already built extraction wrappers. In this chapter we are going to talk about the resilience property against changes to the HTML code. This means, how good is our wrapper to continue extracting the correct data when changes are introduced.

These changes are a problem when using the data extraction tools. If we have a feed that receives information from a concrete Web page and happens that in a certain moment the HTML structure is modified and we can lose the flow of data to this feed. Furthermore, we have to be aware of this situation happens as it is a critical point. We can control it by monitoring the data source and looking that the resulting information is correct. Somehow the detection of problems can be automated by using scripts and also some little programs that test if we receive information or if it follows a certain structure. When a problem appears, some kind of alert could be sent, like a mail to the administrator to fix as faster as possible this problem. What is sure is that we will have to configure again our data extraction tool to have it up to date.

We can classify these changes into categories:

- Changes to the structure of the Web
    - Changing order of the elements
    - Erasing old content
    - Introducing new content
- Changes to the style tags of the Web
- Changes to the visual design of the Web
- Other types of changes

Changes to the Web structure could generate errors depending where our interesting data is placed, e.g. if it is deleted errors will appear. On the other hand, if we introduce new content and it doesn't affect the initial structure of the Web, no problems will appear. Changing order of the elements could introduce errors if our data extraction tool only considers the position of the HTML tags into the parsing tree.

Some tools get information from the DIV or SPAN tags, specifically the class attribute. This helps to locate the data wherever is placed independently of the HTML structure. If the tool uses this information and changes to those tags, which are introduced, we can experience problems.

Changes to the visual design, for example changing the background of the page, from the tables, from the cells or changing font colors will not create errors to our data extraction tools. What commonly happens is that these types of changes are introduced together with changes to the Web structure, and then, we have more possibilities to generate errors.

More errors could be introduced by other types of changes. For example place data that we want to extract into a Flash object or place this data to an emerging Javascript window or into a file available through a link… More examples of this kind can be found.

4.4.1- Testing the resilience of our tools

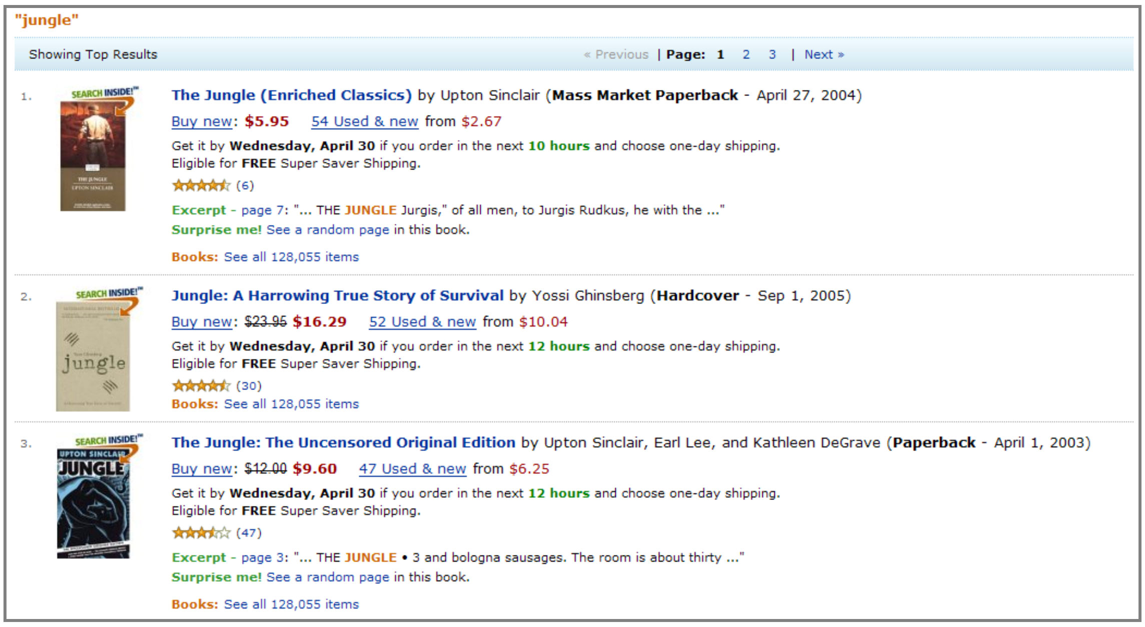
In this section we are going to experiment how a change of the HTML code could affect the correctness of the extracted data by our tools.

We have prepared a stage for this purpose. We have downloaded all the HTML code and required files from a Book search in *Amazon.com*. We have used the input value *Jungle* to perform the search.

Once we have downloaded these files we are going to upload them in a test server. Doing this we assure that the content is going to be static and no new changes are going to be introduced when performing our tests.

By using our tools we are going to extract some fields of this book search, in concrete: title, book format, new price and valuation.

New tests to evaluate resilience will be performed changing the HTML code of this *Amazon* search and replacing the original content. Then we can compare if the extracted data was the same as before or new errors has been generated.


"jungle"

Showing Top Results « Previous | Page: 1 2 3 | Next »

1. The Jungle (Enriched Classics) by Upton Sinclair (Mass Market Paperback - April 27, 2004)
Buy new: $5.95 54 Used & new from $2.67
Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)
Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

2. Jungle: A Harrowing True Story of Survival by Yossi Ghinsberg (Hardcover - Sep 1, 2005)
Buy new: ~~$23.95~~ $16.29 52 Used & new from $10.04
Get it by Wednesday, April 30 if you order in the next 12 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(30)
Books: See all 128,055 items

3. The Jungle: The Uncensored Original Edition by Upton Sinclair, Earl Lee, and Kathleen DeGrave (Paperback - April 1, 2003)
Buy new: ~~$12.00~~ $9.60 47 Used & new from $6.25
Get it by Wednesday, April 30 if you order in the next 12 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(47)
Excerpt - page 3: "... THE JUNGLE • 3 and bologna sausages. The room is about thirty ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

Figure 54: Amazon result page to be used

We have tried to extract all these fields with all of our visual tools but we experienced problems when using some of them. We are going to explain the problems that we have encountered:

- **Automation Anywhere:**

  With this tool we could not extract correctly some fields, for example the book format or the price. Although this is not a problem to evaluate the resilience, this tool only allows the user to extract and save the data. This is a problem when we want to test the resilience property as each time that we want to perform a new extraction we have to select the data and it makes the test not useful, we mean we can not compare two different extractions.

- **WinTask**:

  With this tool we could not extract correctly the fields that we wanted. We have configured it to extract them, but what happened is that the used precision let only to extract the information found on the book cell. For this abscense of precision the resilience test that we are going to apply is not going to be useful and then we are going to rule out this tool to be used with the resilience tests.

- **Goldseeker:**

  Due to the simplicity of this tool it has been not possible to realize the data extraction of all the fields. Only part of the information mixed with another data not of our interest has been extracted. The fact of realizing scripts that work searching for concrete strings creates difficulties to extract concrete data when having a big amount of HTML code. Anyway as we know how this tool works, we can achieve conclusions about its resilience property.

  The fact of selecting two strings that are placed between the content of interest makes this tool resilience in some cases:

- If the content we want to extract is placed between two strings that are not going to be modified then the extraction is going to be performed without problems.

- If the content we want to extract is placed between two strings used for its location and are going to be modified then the extraction is not going to be performed as it is not going to be found and problems will appear.

In conclusion, this tool will have a good resilience property depending on the change realized to the HTML structure. As there are more possibilities to modify other content than the strings used to identify the field, we are going to categorize this tool with a good resilience property.

4.4.2- Structure

If we take a look at the original structure of this Web Page we can see that the content of our interest is located on the second column of the first row of the second table of our HTML code.

We want to extract all the information from the rows of this column. Each one of them represents a publication and we can find all the information that we want to extract. We can make a first test of extracting information using this structure. After that we are going to make modifications on this Web Page that pretend to represent possible changes that a Webmaster could apply to update the content and could lead to data extraction errors.

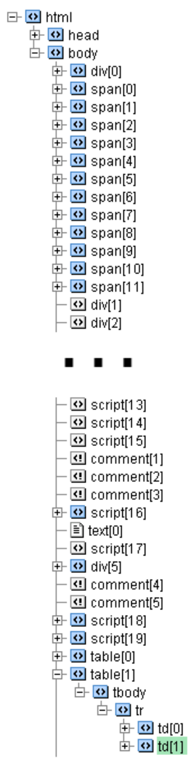
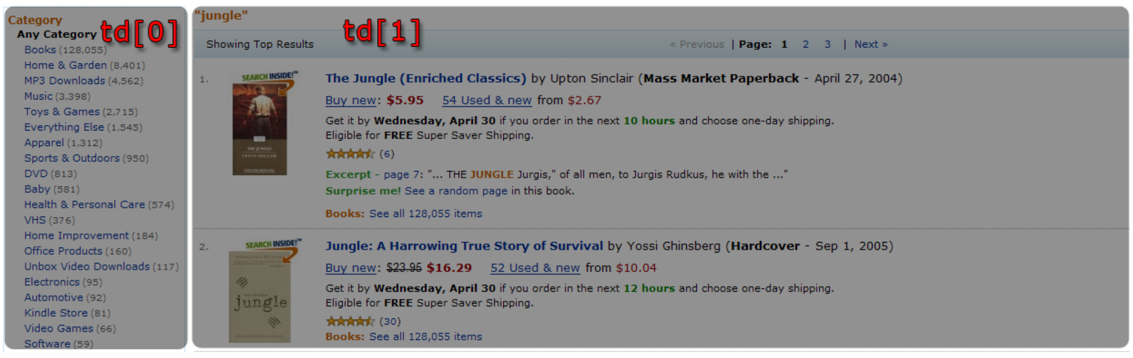


Figure 55: HTML parsing tree structure of the Amazon test

4.4.3- Test 1: Delete a table column next to the extracted data



Category
Any Category
Books (128,055)
Home & Garden (8,401)
MP3 Downloads (4,562)
Music (3,398)
Toys & Games (2,715)
Everything Else (1,545)
Apparel (1,312)
Sports & Outdoors (950)
DVD (813)
Baby (581)
Health & Personal Care (574)
VHS (376)
Home Improvement (184)
Office Products (160)
Unbox Video Downloads (117)
Electronics (95)
Automotive (92)
Kindle Store (81)
Video Games (66)
Software (59)

"jungle"
Showing Top Results « Previous | Page: 1 2 3 | Next »

1. The Jungle (Enriched Classics) by Upton Sinclair (Mass Market Paperback - April 27, 2004)
Buy new: $5.95 54 Used & new from $2.67
Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)
Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

2. Jungle: A Harrowing True Story of Survival by Yossi Ghinsberg (Hardcover - Sep 1, 2005)
Buy new: ~~$23.95~~ $16.29 52 Used & new from $10.04
Get it by Wednesday, April 30 if you order in the next 12 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(30)
Books: See all 128,055 items

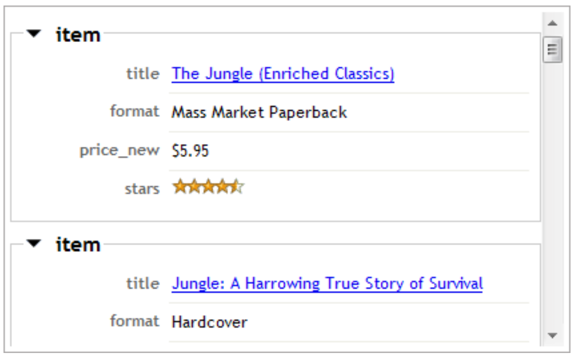
Figure 56: Columns of data for this resilience test

The first attempt to make a modification of the structure of the page consists of deleting the first column of the first row of the second table of the HTML document.

The new column where our interesting data will be placed is the first one. We make a modification using Adobe Dreamweaver to achieve this result.

■ Dapper

We have used the same Dapp as before to extract the same information. This modification in the HTML code doesn't produce any errors. We can assure that dapper is robust to this kind of modifications.



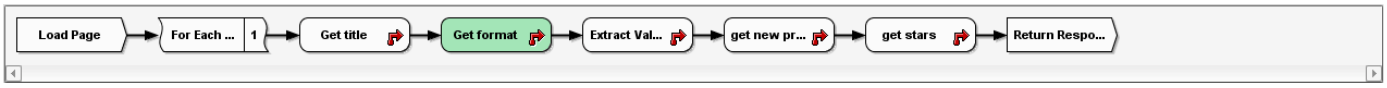
item
title The Jungle (Enriched Classics)
format Mass Market Paperback
price_new $5.95
stars ★★★★☆

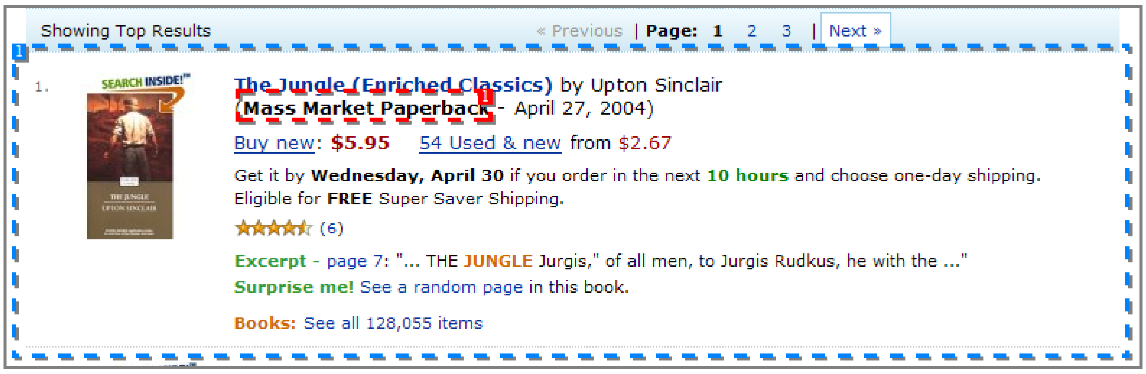
item
title Jungle: A Harrowing True Story of Survival
format Hardcover

Figure 57: Dapper results for this test

■ Robomaker

We don't experience any problem using the same structure as before to extract content, so we can say that Robomaker is robust to this kind of modifications too.




Showing Top Results « Previous | Page: 1 2 3 | Next »

1. The Jungle (Enriched Classics) by Upton Sinclair
(Mass Market Paperback - April 27, 2004)
Buy new: $5.95 54 Used & new from $2.67
Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)
Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

Figure 58: Robomaker results for this test

■ Lixto

With Lixto has happened the same as with our two previous tested tools. We have passed this test and this modification in the HTML code doesn't produce any error.

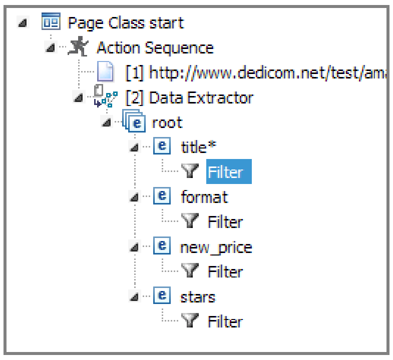


Figure 59: Data model used by Lixto for this test

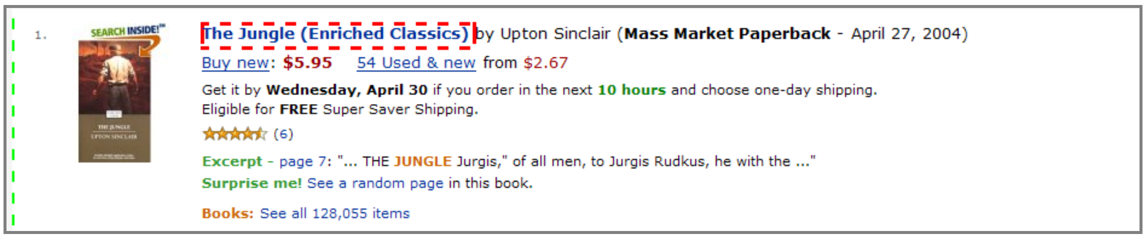

1. The Jungle (Enriched Classics) by Upton Sinclair (Mass Market Paperback - April 27, 2004)
Buy new: $5.95 54 Used & new from $2.67
Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)
Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

Figure 60: Lixto data selection for this test

■ Web Content Extractor

This tool uses an absolute path to identify the elements that appear on the HTML source. We can specify a first data row where the content begins and then, from this point select all the suitable data to be extracted.

As we realized changes that affect to the HTML structure and directly to this path we could not extract information. This tool doesn't therefore pass the test.

■ Webharvest

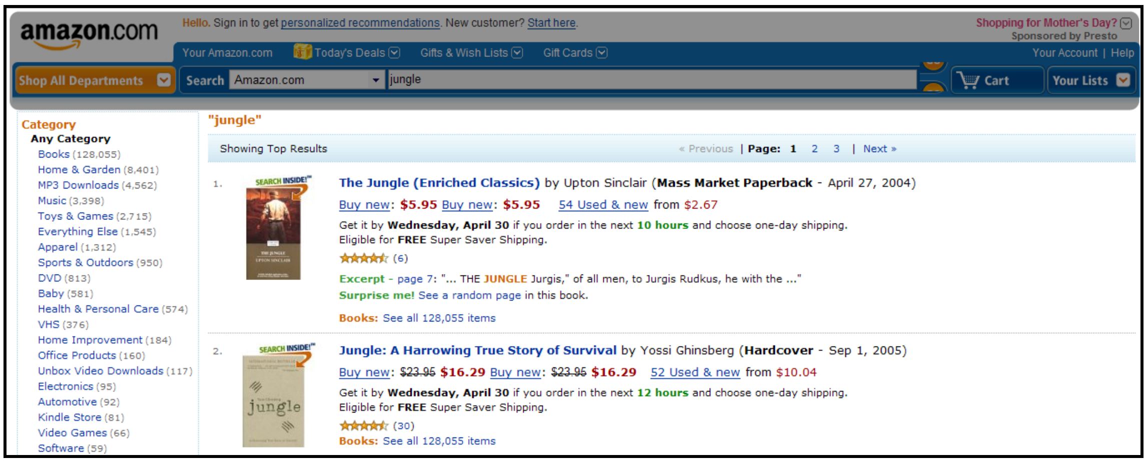
As we are using Xpath expressions to refer to the extracted data fields and we used the information contained in the SPAN tags. This alteration of the content caused no problems when realizing an extraction. Due to this fact Webharvest passed this test.

	1st test of resilience
Dapper	√
Robomaker	√
Lixto	√
Web Content Extractor	X
Webharvest	√

Figure 61: Final test results

4.4.4- Test 2: Delete previous content from the extracted data

The second attempt to make a modification of the structure of the page consists of deleting the first *div* container and a table. Together they represent the dark area of the next image.



Hello. Sign in to get personalized recommendations. New customer? Start here.

Shopping for Mother's Day?
Sponsored by Presto

Category
Any Category
Books (128,055)
Home & Garden (8,401)
MP3 Downloads (4,562)
Music (3,398)
Toys & Games (2,715)
Everything Else (1,545)
Apparel (1,312)
Sports & Outdoors (950)
DVD (813)
Baby (581)
Health & Personal Care (574)
VHS (376)
Home Improvement (184)
Office Products (160)
Unbox Video Downloads (117)
Electronics (95)
Automotive (92)
Kindle Store (81)
Video Games (66)
Software (59)

"jungle"

Showing Top Results « Previous | Page: 1 2 3 | Next »

1. The Jungle (Enriched Classics) by Upton Sinclair (Mass Market Paperback - April 27, 2004)
Buy new: $5.95 Buy new: $5.95 54 Used & new from $2.67
Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)
Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

2. Jungle: A Harrowing True Story of Survival by Yossi Ghinsberg (Hardcover - Sep 1, 2005)
Buy new: ~~$23.95~~ $16.29 Buy new: ~~$23.95~~ $16.29 52 Used & new from $10.04
Get it by Wednesday, April 30 if you order in the next 12 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(30)
Books: See all 128,055 items

Figure 62: Row going to be deleted for this resilience test

As four of our five tested data extraction tools (Dapper, Robomaker, Lixto and Webharvest) passed these tests without problems we are not going to comment each one individually. We can say that all these tools are robust to this kind of modification. With Web Content Extractor happened the same as before, changes to the HTML structure produced problems to realize a successful extraction.

	2nd test of resilience
Dapper	√
Robomaker	√
Lixto	√
Web Content Extractor	X
Webharvest	√

Figure 63: Final test results

4.4.5- Test 3: Making modifications to DIV and SPAN tags

The third attempt to modify the HTML structure consists of making changes at DIV and SPAN tags.

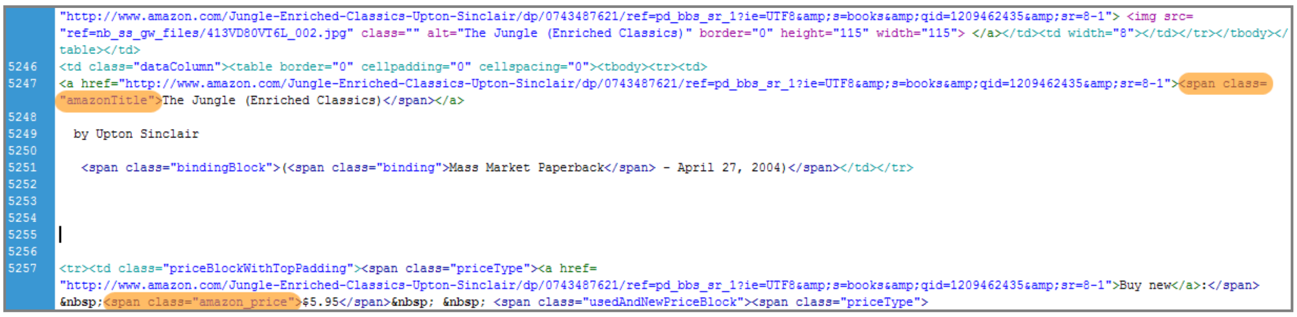
Most of our data extracting tools use them to identify the Web elements we want to extract. We are going to make these two changes:

- Change the class attribute from the *span* tag that identifies the price of a new product.

  The change will be *sr_price* to *amazon_price*.

- Change the class attribute from the *span* tag that identifies the name of a product.

  The change will be *srTitle* to *amazonTitle*.



```
"http://www.amazon.com/Jungle-Enriched-Classics-Upton-Sinclair/dp/0743487621/ref=pd_bbs_sr_1?ie=UTF8&amp;s=books&amp;qid=1209462435&amp;sr=8-1"> <img src=
"ref=nb_ss_gw_files/413VD80VT6L_002.jpg" class="" alt="The Jungle (Enriched Classics)" border="0" height="115" width="115"> </a></td><td width="8"></td></tr></tbody></
table></td>
<td class="dataColumn"><table border="0" cellpadding="0" cellspacing="0"><tbody><tr><td>
<a href="http://www.amazon.com/Jungle-Enriched-Classics-Upton-Sinclair/dp/0743487621/ref=pd_bbs_sr_1?ie=UTF8&amp;s=books&amp;qid=1209462435&amp;sr=8-1"><span class=
"amazonTitle">The Jungle (Enriched Classics)</span></a>

  by Upton Sinclair

   <span class="bindingBlock">(<span class="binding">Mass Market Paperback</span> - April 27, 2004)</span></td></tr>




<tr><td class="priceBlockWithTopPadding"><span class="priceType"><a href=
"http://www.amazon.com/Jungle-Enriched-Classics-Upton-Sinclair/dp/0743487621/ref=pd_bbs_sr_1?ie=UTF8&amp;s=books&amp;qid=1209462435&amp;sr=8-1">Buy new</a>:</span>
&nbsp;<span class="amazon_price">$5.95</span>&nbsp; &nbsp; <span class="usedAndNewPriceBlock"><span class="priceType">
```

Figure 64: HTML code with highlighted Span tags

All the tools except Webharvest were able to extract the book title although we have made a change. That happens because the span tag is inside an <a> tag and this is the one that our tools use to carry out the identification. On the other hand, the Xpath expression of Webharvest used a SPAN tag and we were not able to extract this field.

In that way, all the tools except Web Content Extractor failed when extracting the field price. This happened as they have been not able to relate the old content with the current content.

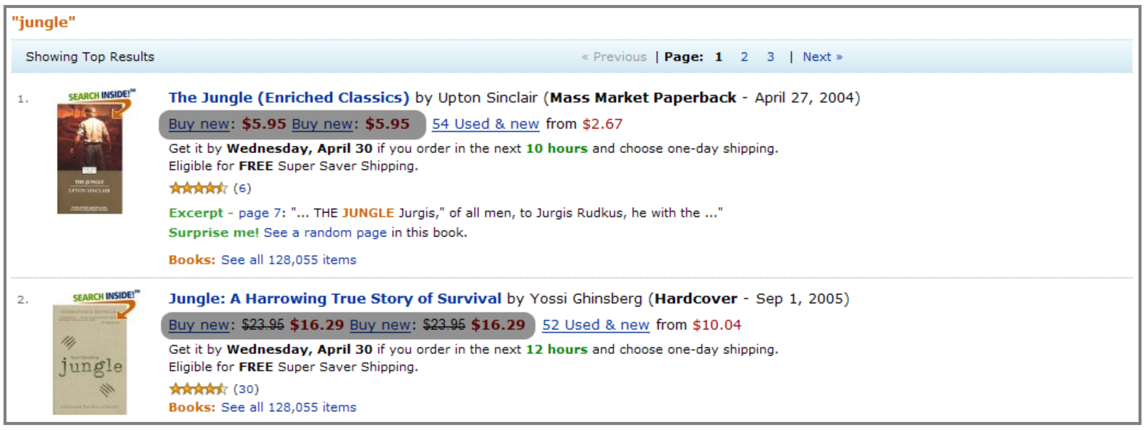
Particularly with Web Content Extractor happened the opposite; it doesn't care about the class attribute of the *id* and *div* tags. It could find all the information and no problems were encountered as the structure remained the same.

	3rd test of resilience
Dapper	√ / X
Robomaker	√ / X
Lixto	√ / X
Web Content Extractor	√
Webharvest	X

Figure 65: Final test results

4.4.6- Test 4: Duplicating extracted data

The fourth attempt to modify the HTML structure consists of duplicating one of the elements that appear and we want to extract. More than a test is a way to see how against these changes our tools react.


"jungle"

Showing Top Results « Previous | Page: 1 2 3 | Next »

1. **The Jungle (Enriched Classics)** by Upton Sinclair (**Mass Market Paperback** - April 27, 2004)
Buy new: **$5.95** Buy new: **$5.95** 54 Used & new from $2.67
Get it by **Wednesday, April 30** if you order in the next **10 hours** and choose one-day shipping.
Eligible for **FREE** Super Saver Shipping.
(6)
Excerpt - page 7: "... THE **JUNGLE** Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.
Books: See all 128,055 items

2. **Jungle: A Harrowing True Story of Survival** by Yossi Ghinsberg (**Hardcover** - Sep 1, 2005)
Buy new: ~~$23.95~~ **$16.29** Buy new: ~~$23.95~~ **$16.29** 52 Used & new from $10.04
Get it by **Wednesday, April 30** if you order in the next **12 hours** and choose one-day shipping.
Eligible for **FREE** Super Saver Shipping.
(30)
Books: See all 128,055 items

Figure 66: Duplicated data for this resilience test

- **Dapper**

It has extracted two times the duplicated new price entry.

- **Robomaker**

It has only extracted one new price entry.

- **Lixto**

It has extracted two times the duplicated new price entry.

- **Web Content Extractor**

It has only extracted one new price entry.

- **Webharvest**

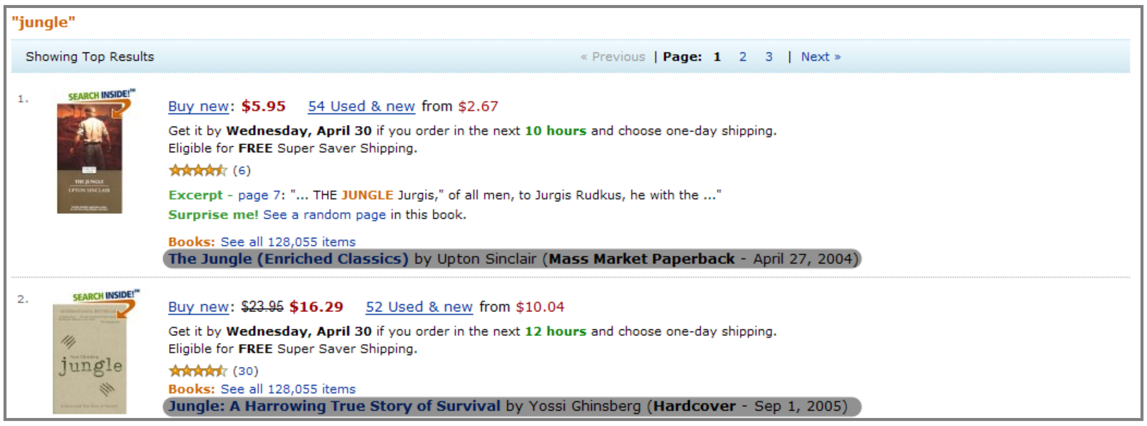
It has extracted two times the duplicated new price entry.

	4th test of resilience
Dapper	Information extracted 2 times
Robomaker	Information extracted 1 time
Lixto	Information extracted 2 times
Web Content Extractor	Information extracted 1 time
Webharvest	Information extracted 2 times

Figure 67: Final test results

4.4.7- Test 5: Changing order of extracted data

The fifth attempt to modify the HTML structure consists of changing the order where data which is going to be extracted appears. In this case we have placed the first row of the two first book entries, where the title and the book format appears, to the last row of the table.


"jungle"

Showing Top Results « Previous | Page: 1 2 3 | Next »

1. Buy new: $5.95 54 Used & new from $2.67

Get it by Wednesday, April 30 if you order in the next 10 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(6)

Excerpt - page 7: "... THE JUNGLE Jurgis," of all men, to Jurgis Rudkus, he with the ..."
Surprise me! See a random page in this book.

Books: See all 128,055 items
The Jungle (Enriched Classics) by Upton Sinclair (Mass Market Paperback - April 27, 2004)

2. Buy new: ~~$23.95~~ $16.29 52 Used & new from $10.04

Get it by Wednesday, April 30 if you order in the next 12 hours and choose one-day shipping.
Eligible for FREE Super Saver Shipping.
(30)
Books: See all 128,055 items
Jungle: A Harrowing True Story of Survival by Yossi Ghinsberg (Hardcover - Sep 1, 2005)

Figure 68: Data order changed for this resilience tests

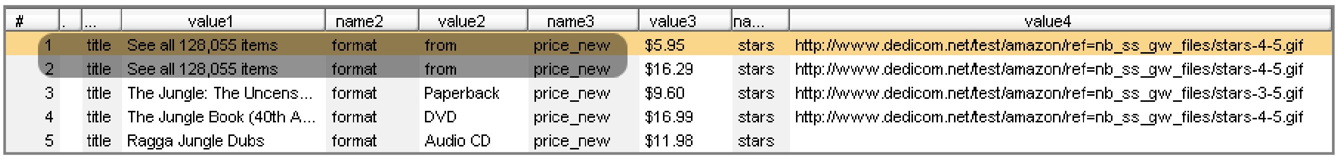
■ Dapper

Dapper displays the following error message so we can not extract data of this modified HTML page.

```
Error While trying to run Events chain : null
```

■ Robomaker

Robomaker has not extracted this two changed fields in a correct way, it has taken information from another rows. Thus, we experience problems because of this type of changes.



#	.	...	value1	name2	value2	name3	value3	na...	value4
1		title	See all 128,055 items	format	from	price_new	$5.95	stars	http://www.dedicom.net/test/amazon/ref=nb_ss_gw_files/stars-4-5.gif
2		title	See all 128,055 items	format	from	price_new	$16.29	stars	http://www.dedicom.net/test/amazon/ref=nb_ss_gw_files/stars-4-5.gif
3		title	The Jungle: The Uncens...	format	Paperback	price_new	$9.60	stars	http://www.dedicom.net/test/amazon/ref=nb_ss_gw_files/stars-3-5.gif
4		title	The Jungle Book (40th A...	format	DVD	price_new	$16.99	stars	http://www.dedicom.net/test/amazon/ref=nb_ss_gw_files/stars-4-5.gif
5		title	Ragga Jungle Dubs	format	Audio CD	price_new	$11.98	stars	

Figure 69: Robomaker results for this test

■ Lixto

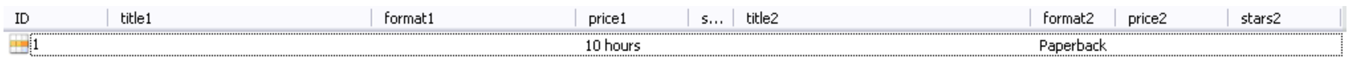
Lixto has extracted all the information without problems. What happened is that we selected the title of the book to be the separator of each group of elements. This means a book entry. Although we have extracted correctly all the data, the structure of the output XML file is not correct. To fix this problem we can select the price to be the separator, but then it will only works applying this change to all the book entries. In conclusion, Lixto doesn't pass this test.

```
<lixto:extractor>
   <root>
      <new_price>$5.95</new_price>
      <stars></stars>
   </root>
```

```
<root>
   <title>The Jungle (Enriched Classics)</title>
   <format>Mass Market Paperback</format>
   <new_price>$16.29</new_price>
   <stars></stars>
</root>
<root>
   <title>Jungle: A Harrowing True Story of Survival</title>
   <format>Hardcover</format>
</root>
<root>
   <title>The Jungle: The Uncensored Original Edition</title>
   <format>Paperback</format>
   <new_price>$9.60</new_price>
   <stars></stars>
</root>
```

■ Web Content Extractor

With Web Content Extractor we could not extract the information due to the changes in the HTML structure. It has extracted wrong data and we can conclude this tool doesn't pass the test.



ID	title1	format1	price1	s...	title2	format2	price2	stars2
1			10 hours			Paperback		

Figure 70: Web Content Extractor results

■ Webharvest

We are using Xpath expressions again to refer to the extracted data fields and we used the information contained in the SPAN tags. This change of order caused no problems when realizing an extraction. Due to this fact Webharvest passes this test.

	5th test of resilience
Dapper	X
Robomaker	X
Lixto	X
Web Content Extractor	X
Webharvest	√

Figure 71: Final test results

4.4.8- A concrete example: Improving resilience with Robomaker against structure changes

Of all of our tools, Robomaker is pretty sure the one with a higher functionality degree. This is the reason why we selected it to improve the resilience of a data extraction of a Web page. To do this we need to go into some of the more advanced and powerful features of this tool.

It is certainly impossible to make a robot that can handle all thinkable and unthinkable scenarios, but a few easy changes can make a robot much more able to handle minor changes such as layout changes or added content.

We are going to create a robot that extracts all the entries of the first two pages of results of *www.digg.com* using an input value to perform a search. The sequence steps to perform the data extraction are the following:

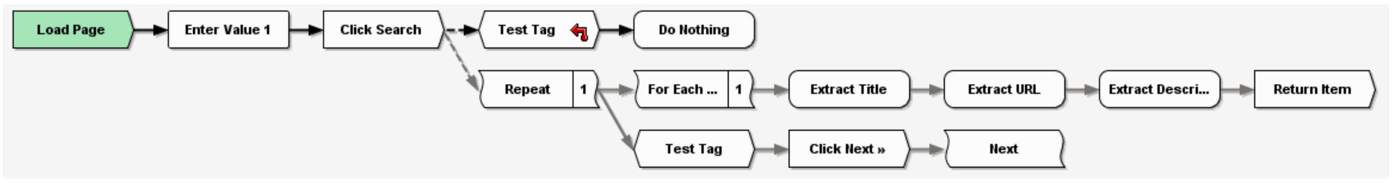


Figure 72: Robomaker step sequence

The robot follows this step sequence:

1- We load the www.digg.com page, we use the input variable to realize the search and click to the search button.
2- Then we use a test tag that uses the following Tag path `.*.div.div.div.div` looking for the class attribute `notice`. If we don't find in the previous defined path such attribute it means that we don't retrieve any result and it has no sense to extract information of the rows.
3- Once we know that results are going to be found we are going to extract the title, the URL and the description of each entry.
4- We are repeating the step 3, for a concrete number of pages by clicking the next button each time. We search the next button by using the following Tag path `.*.div.div.div.div.a` , having the class attribute `nextprev` and using the tag pattern `.*>Next.*` which tests that the text in the link tag starts with the text "Next".

Once we have presented this example we are going to improve its resilience. Let us start by taking a closer look at the Tag finder configuration of the "Test Tag" step.

In this step is the Tag Path `.*.div.div.div` and it doesn't help to the resilience property. This Tag Path points to any Div tag contained inside two other Div tags. If Digg changes its structure and the Div we are looking for is no longer contained within two other Divs then the step will not find this target. If we change the Tag Path manually to `.*.div` we immediately improve the resilience of this step, because now the step is really looking for a Div with the class `notice` anywhere on the page.

The same happens with the next button, we can change the Tag Path to .*.a and then this robot will look for this tag anywhere of the page.

This procedure could be applied to other element steps when constructing our robot, and then we will have a high level of resilience when talking about structure changes. Realizing this kind of changes has consequences. If we could find two next buttons and we are only interested in the one that has the initial Tag Path `.*.div.div.div.div.a` , will make no possible to apply this improvement.

4.5- Precision in extracted data

Another Web data extraction field to consider is the precision that our tools have when extracting data. It means if they can extract the part of the data that we are expecting to. Our data can be structured in several ways; for example it can be placed into a table row, it can be distributed in several table cells or what's more mixed with some other content.

In this section we want to test the accuracy of our tools against this kind of situations. Generally they take benefit from the structure and the information of the HTML tags, but it may happen that the structure of the HTML code could generate problems and the user is not receiving the data that he is expecting.

4.5.1- Precision extracting a date field

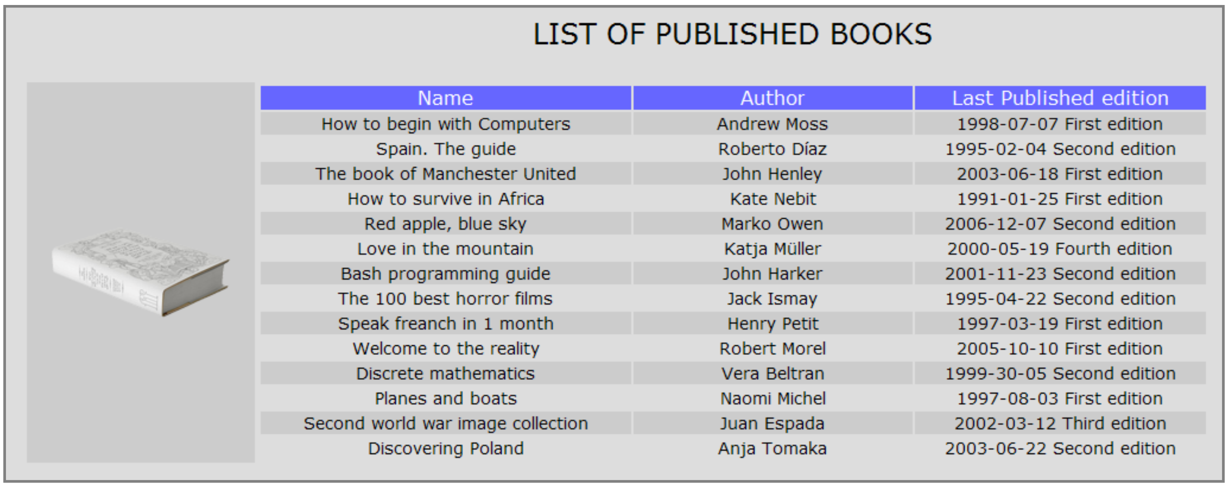
To build a suitable scenario we have designed an HTML page and uploaded it to a server. It consists of a list of books with title, author and the publication date. We are going to extract data from the *Last Published Edition* column. In concrete we are going to extract it with a different precision each time:

- All the information of the row
- Date of the last publication
- Year of the last publication
- 2 last digits numbers of the year of the last publication

This process allows seeing how flexible our tools are to let the user extract information in a more accurate way. In each test we are increasing the acuteness of the extracted data and testing the accuracy property.

4.5.2- Extracting data from simple text

In this first test we are going to construct the HTML source page without using *span* or *div* tags that specifically identify the elements. This is useful as then we can conclude how important is to have this tags in order to identify data elements and extract data from them.



LIST OF PUBLISHED BOOKS

Name	Author	Last Published edition
How to begin with Computers	Andrew Moss	1998-07-07 First edition
Spain. The guide	Roberto Díaz	1995-02-04 Second edition
The book of Manchester United	John Henley	2003-06-18 First edition
How to survive in Africa	Kate Nebit	1991-01-25 First edition
Red apple, blue sky	Marko Owen	2006-12-07 Second edition
Love in the mountain	Katja Müller	2000-05-19 Fourth edition
Bash programming guide	John Harker	2001-11-23 Second edition
The 100 best horror films	Jack Ismay	1995-04-22 Second edition
Speak freanch in 1 month	Henry Petit	1997-03-19 First edition
Welcome to the reality	Robert Morel	2005-10-10 First edition
Discrete mathematics	Vera Beltran	1999-30-05 Second edition
Planes and boats	Naomi Michel	1997-08-03 First edition
Second world war image collection	Juan Espada	2002-03-12 Third edition
Discovering Poland	Anja Tomaka	2003-06-22 Second edition

Figure 73: First constructed scenario for the precision tests

The final results for each tool can be found in the next table:

	All the information of the row	Date of the last publication	Year of the last publication	2 last digits of the year of the last publication
Dapper	√	X	X	X
Robomaker	√	√	√	√
Lixto	√	√	√	√
WinTask	√	X	X	X
Automation Anywhere	√	X	X	X
Web Content Extractor	√	√	√	√
Goldseeker	√	√	√	√
Webharvest	√	X	X	X

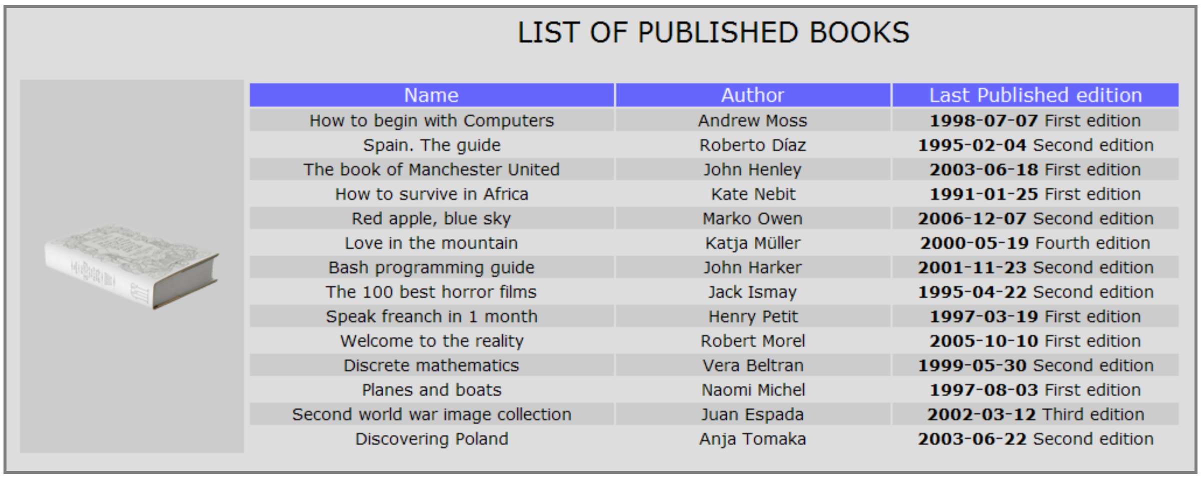
Figure 74: Final results extracting data from simple text

The difference using Lixto, Robomaker and Web Content Extractor compared with the other tools is that we can use extra features that allow us to extract data in a more accurate way. This means applying a concrete format to the data or realizing transformations of it before getting the final content. The substring method of Goldseeker is useful when modifying the precision. As shown in the table this four tools passed all the tests.

4.5.3- Extracting data from formatted text

In this second test we are going to highlight the date of the *Last Published* edition field applying a bold style.

With this process we are placing a *<strong>* tag into our HTML code and this will help our tools to distinguish among the two parts of the *Last Published edition* field.



LIST OF PUBLISHED BOOKS

Name	Author	Last Published edition
How to begin with Computers	Andrew Moss	**1998-07-07** First edition
Spain. The guide	Roberto Díaz	**1995-02-04** Second edition
The book of Manchester United	John Henley	**2003-06-18** First edition
How to survive in Africa	Kate Nebit	**1991-01-25** First edition
Red apple, blue sky	Marko Owen	**2006-12-07** Second edition
Love in the mountain	Katja Müller	**2000-05-19** Fourth edition
Bash programming guide	John Harker	**2001-11-23** Second edition
The 100 best horror films	Jack Ismay	**1995-04-22** Second edition
Speak freanch in 1 month	Henry Petit	**1997-03-19** First edition
Welcome to the reality	Robert Morel	**2005-10-10** First edition
Discrete mathematics	Vera Beltran	**1999-05-30** Second edition
Planes and boats	Naomi Michel	**1997-08-03** First edition
Second world war image collection	Juan Espada	**2002-03-12** Third edition
Discovering Poland	Anja Tomaka	**2003-06-22** Second edition

Figure 75: Second constructed scenario for the precision tests

The resulting table is the following:

	All the information of the row	Date of the last publication	Year of the last publication	2 last digits of the year of the last publication
Dapper	√	√	X	X
Robomaker	√	√	√	√
Lixto	√	√	√	√
WinTask	√	X	X	X
Automation Anywhere	√	X	X	X
Web Content Extractor	√	√	√	√
Goldseeker	√	√	√	√
Webharvest	√	√	X	X

Figure 76: Final results extracting data from formatted text

The only column that has changed is the second one as now some of our programs could split up the content taking advantage of the *<strong>* tag.

4.5.4- Extracting data using styled text

It is very common to use CSS to define a style to our text. We can use a CSS style to identify the elements that appear in the Web taking the information of the attribute *class*.

For example we are going to format only the date from the *Last Published* edition field using the following CSS entry:

```
.date {font-family: Verdana, Arial, Helvetica, sans-serif; font-size:
14px; }
```

Then, the data HTML source would be probably as follows:

```
<div align="center" class="Estilo9">
  <span class="date">
    1998-07-07
  </span>
  First edition
</div>
```

With this kind of tagging our tools could recognize the date and separate it from the entire field.

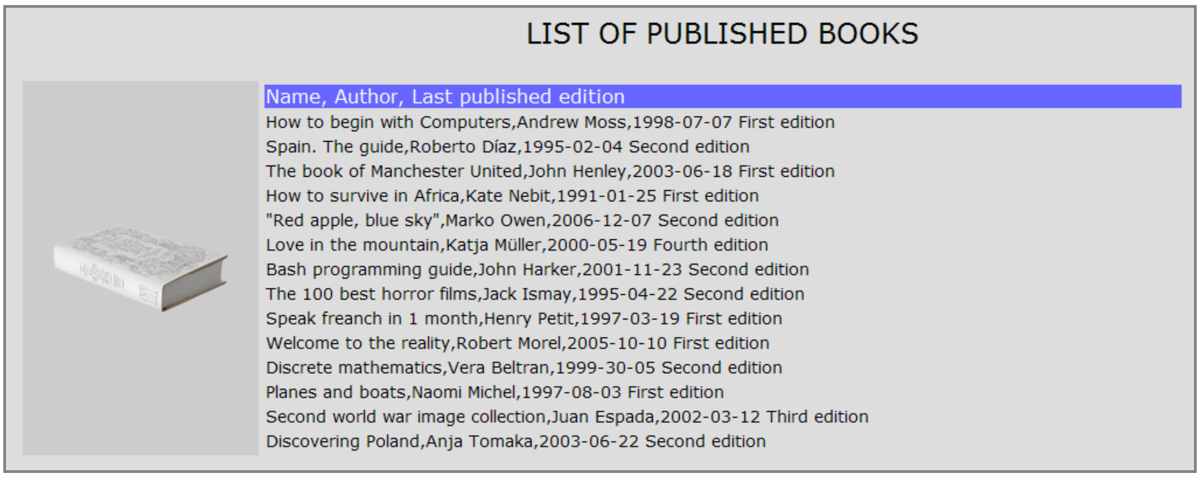
The resulting table is the following:

	All the information of the row	Date of the last publication	Year of the last publication	2 last digits of the year of the last publication
Dapper	√	√	X	X
Robomaker	√	√	√	√
Lixto	√	√	√	√
WinTask	√	X	X	X
Automation Anywhere	√	X	X	X
Goldseeker	√	√	√	√
Webharvest	√	√	X	X

Figure 77: Final results from extracting data using styled text

4.5.5- Extracting data from CSV formatted text

Now we are going to try it with CSV data. It is a file type that stores tabular data and uses a comma to separate values. We are going to place all the information in the same column and separate the fields with commas. This new HTML page will look like this one:


LIST OF PUBLISHED BOOKS

Name, Author, Last published edition
How to begin with Computers,Andrew Moss,1998-07-07 First edition
Spain. The guide,Roberto Díaz,1995-02-04 Second edition
The book of Manchester United,John Henley,2003-06-18 First edition
How to survive in Africa,Kate Nebit,1991-01-25 First edition
"Red apple, blue sky",Marko Owen,2006-12-07 Second edition
Love in the mountain,Katja Müller,2000-05-19 Fourth edition
Bash programming guide,John Harker,2001-11-23 Second edition
The 100 best horror films,Jack Ismay,1995-04-22 Second edition
Speak freanch in 1 month,Henry Petit,1997-03-19 First edition
Welcome to the reality,Robert Morel,2005-10-10 First edition
Discrete mathematics,Vera Beltran,1999-30-05 Second edition
Planes and boats,Naomi Michel,1997-08-03 First edition
Second world war image collection,Juan Espada,2002-03-12 Third edition
Discovering Poland,Anja Tomaka,2003-06-22 Second edition

Figure 78: Fourth constructed scenario for the precision tests

In the following, we are going to use our tools to extract the same content as before. Tools with data transformation and more extraction accuracy have better extracted our desired information.

	All the information of the last published edition	Date of the last publication	Year of the last publication	2 last digits of the year of the last publication
Dapper	X	X	X	X
Robomaker	√	√	√	√
Lixto	√	√	√	√
WinTask	X	X	X	X
Automation Anywhere	X	X	X	X
Web Content Extractor	√	√	√	√
Goldseeker	√	√	√	√
Webharvest	X	X	X	X

Figure 79: Final results from extracting data from CSV formatted text

5- Concatenating the input/output of our tools

All of our HTML-aware data extraction tools produce an output once the data has been extracted. This output could be given in several formats but from all of them is really interesting to realize that we can reutilize the outputted data in HTML again for the input of our programs.

Such a feature could be useful to extract some part of the data which produces problems with one of these tools; we can use a tool to extract some part of the data and other tool to extract all the other part. Another useful characteristic of this process is that we can separate the extraction process in steps increasing the precision of the extracted data each time.

If we take a look to the table of our data tools features we can see that Dapper, Roadrunner and Web Content Extractor can use the HTML format both for input and output.

In this chapter we are going to carry out some tests combining two of these tools to see if this process could fix some problems in the data extraction process or might be useful to have several precision of the extracted data.

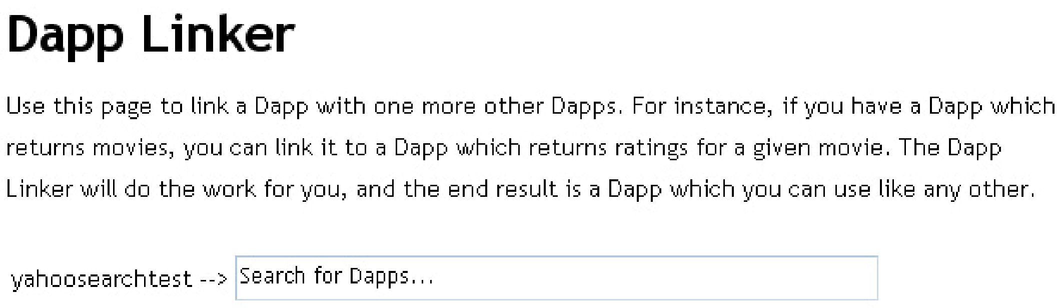
We are going to realize the following combinations with our programs:

- Dapper to Dapper
- Dapper to Web Content Extractor
- Web Content Extractor to Dapper
- Web Content Extractor to Web Content Extractor

5.1- Dapper to Dapper

When talking about using the output of one Dapp to the input of another Dapp, we can already use a built option from Dapper: it is the Dapp Linker.

To use it we only have to select a first built Dapper that is going to provide us an output and then select another one that is going to use this data as input.



Dapp Linker

Use this page to link a Dapp with one more other Dapps. For instance, if you have a Dapp which returns movies, you can link it to a Dapp which returns ratings for a given movie. The Dapp Linker will do the work for you, and the end result is a Dapp which you can use like any other.

yahoosearchtest --> Search for Dapps...

Figure 80: Dapp linker screenshot

Our first Dapp was configured to extract data from a previous example of our documentation; it was the *Kings of Sun 2008 Contest* of Chapter 2.4.

We have configured this Dapp to extract almost all of the data given.

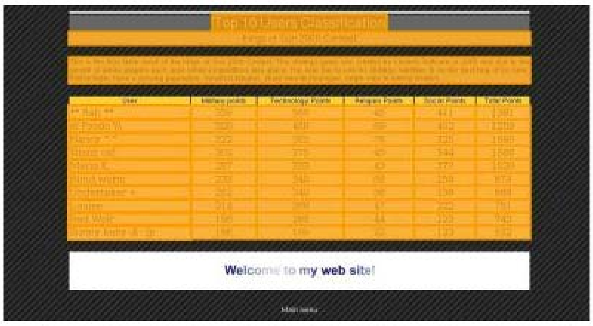
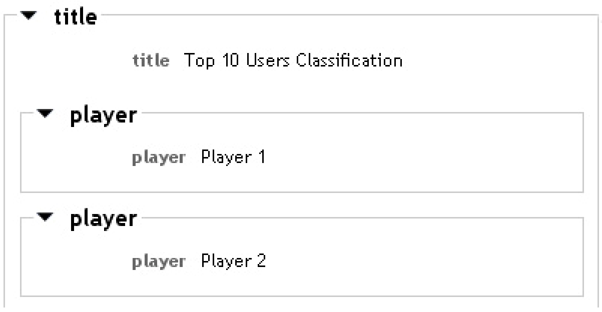


Figure 81: Highlighted data extracted by the first Dapp

A second Dapp has been created to extract data from the same source. But this time we only have extracted the name of the players.

By linking these two wrappers we obtained a new output. The format of this output is constructed having for each entry of the first wrapper all of the entries of our second wrapper.



title

title Top 10 Users Classification

player

player Player 1

player

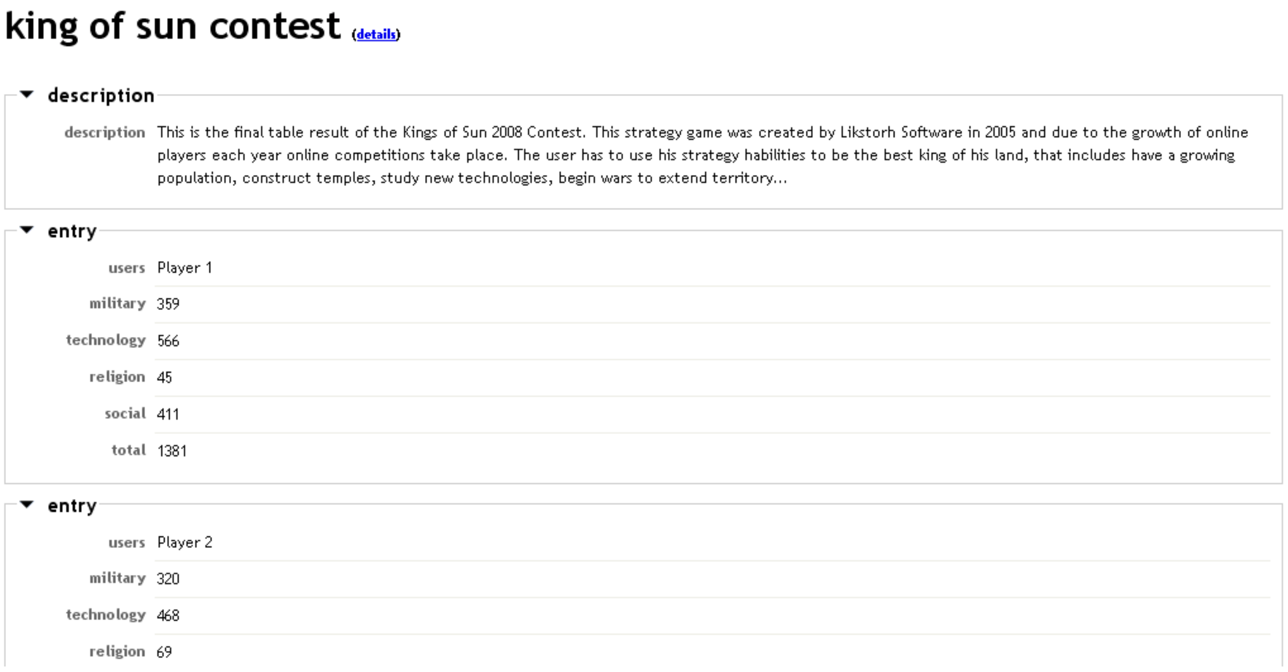
player Player 2

Figure 82: Dapp linker final results

5.2- Dapper to Web Content extractor

In this case we, are going to combine the output and the input of two different tools. One of the output formats of Dapper is HTML, we are going to take advantage of this feature to use this resulting HTML code as the input of the Web Content Extractor tool.

A part of the output given by Dapper is the following:



king of sun contest (details)

description

description This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each year online competitions take place. The user has to use his strategy habilities to be the best king of his land, that includes have a growing population, construct temples, study new technologies, begin wars to extend territory...

entry

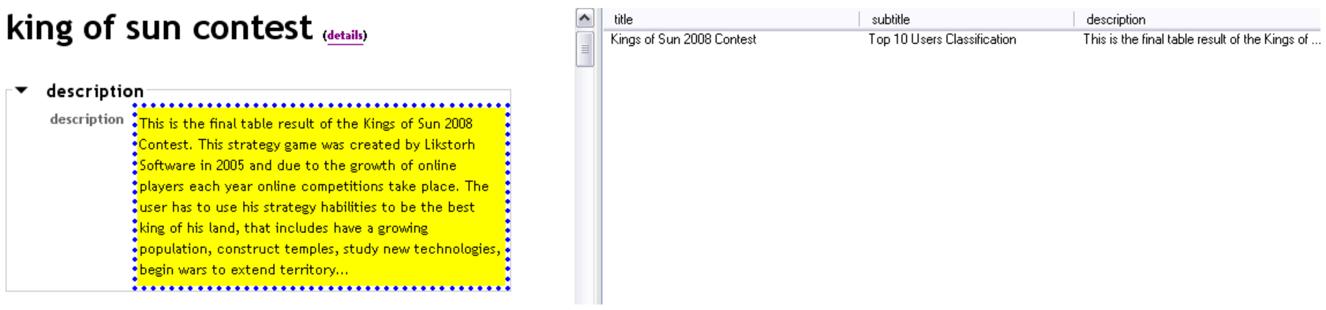
users	Player 1
military	359
technology	566
religion	45
social	411
total	1381

entry

users	Player 2
military	320
technology	468
religion	69

Figure 83: Dapper HTML output

After that we are going to use this output with Web Content Extractor. We are going to extract the title, the subtitle and the description of this Web site.



king of sun contest (details)

description

description This is the final table result of the Kings of Sun 2008 Contest. This strategy game was created by Likstorh Software in 2005 and due to the growth of online players each year online competitions take place. The user has to use his strategy habilities to be the best king of his land, that includes have a growing population, construct temples, study new technologies, begin wars to extend territory...

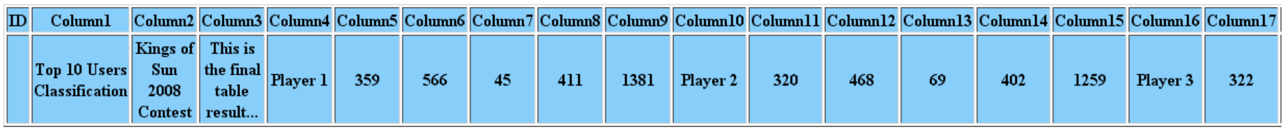
title	subtitle	description
Kings of Sun 2008 Contest	Top 10 Users Classification	This is the final table result of the Kings of ...

Figure 84: Web Content Extractor final results

As shown in the previous figure Web Content Extractor has realized a successful data extraction from the output of Dapper. In conclusion we can concatenate the results of Dapper to the input of Web Content Extractor.

5.3- Web Content Extractor to Dapper

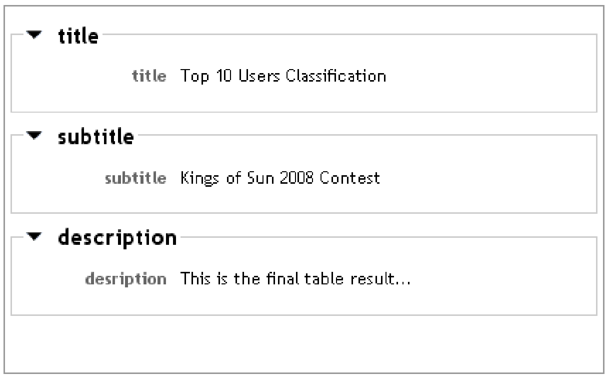
In this case the first program to produce the first output will be Web Content Extractor and the tool to receive the input Dapper. Let's configure WCE, we are going to extract almost all of the data of our *Kings of Sun 2008 Contest.*



ID	Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8	Column9	Column10	Column11	Column12	Column13	Column14	Column15	Column16	Column17
	Top 10 Users Classification	Kings of Sun 2008 Contest	This is the final table result...	Player 1	359	566	45	411	1381	Player 2	320	468	69	402	1259	Player 3	322

Figure 85: Web Content Extractor HTML output

Once we receive all the results in HTML we are going to use Dapper to extract some of these fields. Configuring it to extract the title, the subtitle and the description will not generate problems and the output will look like that:



title

title Top 10 Users Classification

subtitle

subtitle Kings of Sun 2008 Contest

description

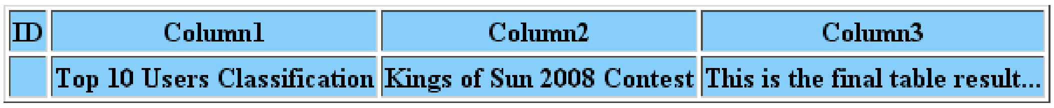
desription This is the final table result...

Figure 86: Dapper final results

5.4- Web Content extractor to Web Content extractor

In this last case we are going to use Web Content Extractor twice to extract information and link the output to the input.

The first HTML output data looks like Figure 85 from the previous case and the final output is really similar as we are only extracting some fields of all the information.



ID	Column1	Column2	Column3
	Top 10 Users Classification	Kings of Sun 2008 Contest	This is the final table result...

Figure 87: Web Content Extractor final results

6- Categorization of the data extraction tools

This chapter deals with about the necessity of categorize the data extraction tools. It can be seen as a final conclusion after realizing tests, achieve results, read documentation and all the work done in this project.

The goal of the data extraction tools categorization is to give an idea to the final user to know in what kind of scenario one tool is better than another one, which are the advantages and disadvantages of each tool and to realize a final conclusion analyzing several characteristics. We also want to give too a qualitative approach of some of the characteristics that our tools have.

It is true that tools having a GUI give facilities to the user compared to the non-GUI tools. Anyway, is really useful to analyze all the cases. Let's present an example: We want to extract certain data from a Web page and we want the output in a concrete format, for an enterprise no problem will occur, the license can be bought and a GUI tool will be comfortable. For an individual user could be too expensive to buy a license to realize only few or concrete data extractions. For this user a free non GUI tool will be better.

A categorization of the tools is going to be constructed considering qualitative characteristics derived from the tests conducted in this document. This process make us able to realize conclusions and to select which the best types of scenarios for each tool, knowing what the strong and weak points are.

A table is presented as follows. It has a set of features (some obtained from the own tool and others obtained through our tests execution) that will help us to carry out the final categorization.

Legend:

Poor/Low	--	-	0	+	++	Good/High

NR = No Result

	Ease of use	Resilience	Web Search Engines / Ebay	Dynamic Content	Precision	Output Formats	Impression	Total
Dapper	++	+	+	-	-	++	++	+
Robomaker	0	+	++	-	++	+	++	++
RoadRunner	--	NR	NR	NR	NR	0	0	0
XWRAP	-	NR	NR	NR	NR	0	0	0
Lixto	0	+	++	-	++	0	++	+
WebHarvest	-	++	++	--	-	0	-	0
GoldSeeker	-	+	--	--	++	-	-	-
WinTask	0	NR	--	-	--	0	0	-
Automation Anywhere	++	NR	--	-	--	+	0	-
Web Content Extractor	+	-	++	-	++	++	+	+

Figure 88: Tools categorization using qualitative features

Once we obtained this information we are able to group our tools and analyze the features that can be used in a concrete scenario. This is the main goal of realizing a categorization as this helps us to select the best tool knowing the strong and weak points of it.

General features like the complexity, ease of use or the output formats have been used here as they are relevant when expecting results from our tools. On the other hand information from our realized tests has been used to fill other columns: Resilience, Web Search Engines and Ebay and dynamic content and precision.

The column impression includes other general aspects not shown in the other columns like installation, configuration, presentation, number of options…

Using this table and having knowledge of each of the tools we are going to start with the categorization:

- **XWRAP** – Non GUI tools without editing files

In this group of tools we can find tools which don't use any GUI and don't use any configuration file. We have to say that is not totally true as we use an Internet browser to realize the configuration of the data extraction process but we can consider this not a truly GUI as we only feel forms and use buttons to send orders.

In concrete, XWRAP is used to realize a sequence of steps that allow us to configure the data extraction process. In each step we configure concrete characteristics like elements identification, data tagging, refinements…

Although we can not realize executions due to the library support, the recommended scenario to use this tool is the set of Web pages with a simply structure. When referring to simple structure we mean no dynamic content, a logic structure and no input variables. It is mainly designed to extract data from plain HTML files.

- **Roadrunner** – Non GUI tools with configuration files

In this group of tools we can find tools that don't use a GUI but they take the input of configuration files to work. This way to proceed is more logic and common than the previous one as the fact of no having a GUI force us to feed the input of our tools with configuration files.

Roadrunner belongs to this group of tools. We only use configuration files and the Linux shell to proceed with the data extraction process.

Although we have not used it in our tests the recommended scenario is similar to the one that we have with XWRAP. This means that we have to select a kind of basic pages that allow us to extract information from them. The fact of only using a specific configuration file makes sometimes difficult to configure the data extraction process.

- **Web-Harvest** – Non GUI tools with XPath expressions

In this group of tools we can find Web-Harvest which is characteristic for the fact that it uses Xpath expressions to extract data.

XPath expressions are quite common and it is really easy to find information for its usage. Sometimes it can be difficult to find the correct expression to extract data and maybe we have to concatenate several of them. It may also happen that we can not find a suitable expression to extract data of our interest.

Web-Harvest is categorized in this group of tools. To realize a data extraction process we have to look at the user manual found in the project Homepage and find all the possible methods and expressions that can be used to carry out concrete actions.

As in the two previous groups of tools it is recommended to realize data extractions in basic scenarios. However, it is also true too that by realizing more complex configurations we can extract information from non basic Web sites.

- **Goldseeker** – Non GUI tools with own scripting

This group of tools is really similar to the one using configuration files. The difference consists of using scripts in the place of several lines containing configuration methods.

Goldseeker is from all of the tools, the more basic one and in an earlier phase of development. The way to realize the data extraction process is reduced on editing a small configuration file with some of the commands that can be found in the readme.txt file.

In conclusion, we don’t recommend the use of this tool to perform data extractions as we could not realize in some cases basic extractions. It is a kind of basic development tool that was interesting to test but it doesn’t have a real utility when expecting professional results.

When we realize a final conclusion of the entire group we conclude the same as in the two first groups, we can only perform extractions using simple structure Web pages.

- **Dapper, Automation Anywhere** – GUI tools without using scripts or expressions

The main characteristic of this group is the facility to realize a configuration for the data extraction process. The fact of having a GUI and don’t use scripts or expressions turn the process to an easy sequence of steps that a non advanced user can do. On the other hand the weak point of the tools is that they don’t use scripts or expressions. Due to this fact, advanced data extractions can not be realized but normal or complete data extractions are a real possibility.

We can find two tools in this group, although they have similarities they have at the same time important differences. Dapper is more complete as it is only focused on the data extraction process and has a lot of output formats. Talking about Automation Anywhere it can realize other types of tasks but only simple data extractions. From these two tools the use of Dapper is recommended.

- **Winstask, Web Content Extractor** – GUI tools using scripts or expressions and without full data extraction support

In this group we can find tools using a GUI and expressions or scripts to perform extractions.

The fact of using scripts and expressions give these tools more chances and possibilities to extract concrete information but at the same time introduce a more complex process for the data extraction.

Compared to the previous group, they are more suitable to be used in professional areas as we can extract information in a more precise way. The main problem of these tools is that they are not constructed to be focused only on the data extraction process; they are built to automate tasks and other type of stuff and often more functionalities are required. Wintask and Web Content Extractor belong to this group and they are built for the Windows operating system.

- **Robomaker, Lixto** – GUI tools using scripts or expressions and with full data extraction support

This last group of tools use a GUI, use scripts and expressions and have a fully data extraction support. For this reason, it is not difficult to conclude this kind of tools are the most powerful ones and recommended to realize all type of data extractions. Robomaker and Lixto belong to this group.

We can carry out this process from the most simple data extractions to the most complex ones. It is true that at the beginning we have to learn how to use them and to realize complex data extractions, which means spend some time experiencing with the tools. In conclusion, they are in a general view the best professional option to extract data from Web sites.

7- Conclusions

In this last chapter we write conclusions of all the entire developed work, pointing out the most important features for a data extraction tool taking care of some criterias and user profiles.

Next we explain the problems that we have faced doing all the tests, documentation and other general aspects.

To conclude, possible future work and some ideas to go further with this project are explained to finalize this document.

From all the features analyzed in the chapter 3.4, we are going to explain which the most important are and why. In fact, all are really useful but due to the scenarios that we can find on the Web and due to the usability some stand out.

- **Interface:** The fact of using a GUI to accomplish extractions give the user a high degree of ease when configuring and performing the data extractions. As a matter of fact it is normal that when extracting data from visual sources, like Web sites, the use of a GUI become a natural and logic way to treat with the content and to select the sources. The absence of a GUI makes the things harder as we don't have a direct relation between the content that we see and the HTML code or configuration files that we treat with.

- **Engine:** Another really important point to consider is the used engine when realizing data extractions. Some tools base their extractions direct on the parsing tree derived from the HTML sources, other ones take care of the tag type where the data of our interest is placed and others use alternative methods like Goldseeker and its substring system. In conclusion, the best results are achieved when having a hybrid engine of these explained methods as this really helps to increase the level of resilience.

- **Scripts and expressions:** As explained on the last chapter the fact of using scripts and expressions give the tools more chances and possibilities to extract and to treat concrete information. Tools having this feature will be more powerful when executing data extractions.

On the other hand it is true that depending on the user profile and the data extraction needs one tool will be more suitable than another. Considering that a final user profile is a single user or an enterprise (or researching group) and taking care of the complexity of the extractions and the price of the license, we are going to construct a table. The presented order of the tools is used to give priority to the best ones.

Single user		Enterprise or researching group	
Basic extractions	Complex extractions	Basic extractions	Complex extractions
1- Dapper 2- Webharvest 3- Goldseeker	1- Robomaker	1- Web Content Extractor 2- Dapper 3- Webharvest 4- Wintask 5- Goldseeker 6- Automation Anywhere	1- Robomaker 2- Lixto

Figure 89: Tools categorization using user profiles

About the problems that we encountered developing all the work, we could mention some installation and configuration difficulties at the beginning, especially with some Linux tools. Some of the tools had a trial-version license and we had to execute all the tests within the license time period. In fact, once we had to reinstall all the entire operating system to continue using these tools.

Some of the non-GUI tools need a higher level of configuration and the time spent to configure them was quite bigger than the GUI tools. As also explained in the chapter 4.3 we could not execute our tests with XWRAP and Roadrunner.

Doing a global critique of our tools, we emphasize the reality that none of our tool could achieve good extractions from dynamic content pages. This is an important point, as more and more dynamic content is introduced nowadays and the use of AJAX and Javascript technologies becomes a usual fact.

Finished this job we let the way open to go further with future work. The execution of more tests taking more scenarios could give more precision to our conclusions. On the other hand, increasing the number of the data extraction tools allows extending the information that we have elaborated in this document and to expand the set of tools. Also taking tools from other taxonomies can widen the final conclusions and the tests done in this document.

8- References

[1] Amazon
http://www.amazon.com

[2] Cyberneko HTML Parser homepage
http://sourceforge.net/projects/nekohtml

[3] Ebay
http://www.ebay.com

[4] Google
http://www.google.com

[5] Html2xhtml
http://www.it.uc3m.es/jaf/html2xhtml

[6] Microsoft Live Search
http://www.live.com/

[7] Netcraft Ltd
http://www.netcraft.com

[8] Pageflakes
http://www.pageflakes.com

[9] Searchenginewatch.com
http://searchenginewatch.com

[10] Wikipedia, the free Encyclopedia
http://en.wikipedia.org

[11] World Wide Web Consortium.
http://www.w3.org

[12] Yahoo! Search
http://search.yahoo.com/

[13] Abiteboul S: Querying Semi-Structured Data. ICDT 1997

[14] Ashish, N.; Knoblock, C.A.: Wrapper Generation for Semi-structured Internet Sources. SIGMOD Record 26(4), 1997

[15] Aumüller D.; Thor A.: Mashup-Werkzeuge zur Ad-hoc Datenintegration im Web, 2008

[16] Baumgartner R.; Ceresna M.; Ledermüller G.: DeepWeb Navigation in Web Data Extraction. CIMCA/IAWTIC 2005

[17] Baumgartner R.; Fresca S.; Gottlob G.: Visual Web Information Extraction with Lixto. VLDB 2001

[18] Chang C.; Kayed M.; Girgis M. R., Shaalan K. F.: A Survey of Web Information Extraction Systems. IEEE Trans. Knowl. Data Eng. (TKDE) 18(10), 2006

[19] Crescenzi V.; Mecca G.; Merialdo P.: RoadRunner: Towards Automatic Data Extraction from Large Web Sites. VLDB 2001

[20] Eikvil L.: Information extraction from world wide web - a survey. Technical Report 945, Norwegian Computing Center, 1999

[21] Fiumara G.: Automated Information Extraction from Web Sources: a Survey, Between Ontologies and Folksonomies Workshop in 3rd International Conference on Communities and Technology, 2007

[22] Hammer J.; Garcia-Molina H.; Cho J.;Aranha R.; Crespo A.: Extracting Semistructured Information from the Web. Workshop on management of semistructured data, 1997

[23] Kuhlins S.; Tredwell R.: Toolkits for Generating Wrappers. NetObjectDays, 2002

[24] Laender A. H. F.; Ribeiro-Neto B. A.; da Silva A. S.; Teixeira J. S.: A Brief Survey of Web Data Extraction Tools. SIGMOD Record (SIGMOD) 31(2), 2002

[25] Liu B.: WWW-2005 Tutorial: Web Content Mining. Fourteenth International World Wide Web Conference (WWW-2005), 2005

[26] Liu B.; Grossman R.; Zhai Y.: Mining Data Records in Web Pages. KDD 2003

[27] Meiling S.; Kempe C.: Vergleich von IE-Werkzeugen, 2008

[28] Myllymaki J.: Effective Web Data Extraction with Standard XML Technologies. Computer Networks (CN) 39(5), 2002

9- Tools

[29] Dapper Homepage
http://www.dapper.net

[30] Lixto Software GmbH Homepage
http://www.lixto.com

[31] Openkapow Homepage
http://openkapow.com/

[32] Wintask
http://www.wintask.com

[33] Automation Anywhere
http://www.tethyssolutions.com/automation-software.htm

[34] Web Content Extractor
http://www.newprosoft.com/Web-content-extractor.htm

[35] The Roadrunner project Homepage
http://www.dia.uniroma3.it/db/roadRunner/

[36] XWRAP Elite Homepage
http://www.cc.gatech.edu/projects/disl/XWRAPElite/

[37] Web-Harvest Homepage
http://Web-harvest.sourceforge.net/

[38] Goldseeker Project Homepage
http://goldseeker.sourceforge.net/

10- Declaration of authorship

"Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann".

Ort	Datum	Unterschrift